Nr. 3/87 März
DM 6.50, sfr 6.50, öS 50, Lit 5900, hfl 7.50

peeker

MAGAZIN FÜR MIKROCOMPUTER

DHGR-Grafik-Paket

Aztec-C-Compiler

Logik-Analysator

65C816 für IIc

Adreßverwaltung

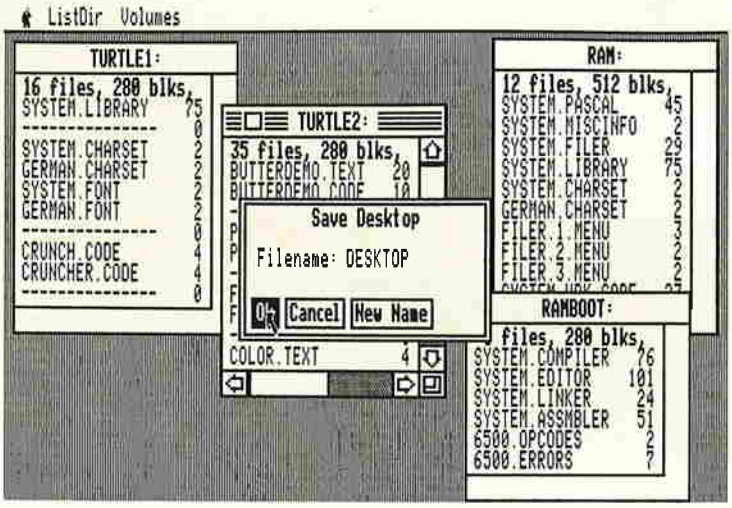
CAD-System

Hüthig
PUBLIKATION

TurtleGraphics-Library-Paket

von Dieter Geiß

Turtle-Utilities für Fenstertechnik und Apple-Maus in einfacher und doppelter Hires-Grafik für Pascal 1.1/1.2 auf Apple II+/e/c mit Maus oder Joystick.
2 Disketten mit umfangreichem Manual, DM 98,–



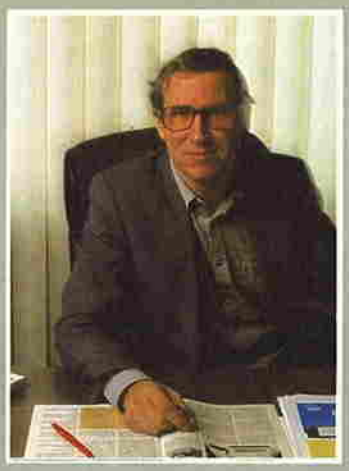
Das Utility-Paket besteht aus vier Modulen, die von Programmierern benutzt werden können, um professionelle grafische Anwendungsprogramme in Pascal zu schreiben.

Benötigt wird ein Apple Pascal Betriebssystem, entweder die Version 1.1 oder die neue Version 1.2. Bestehende Programme laufen ohne Einschränkung mit der neuen „TurtleGraphics", wenn diese nicht zu viel Speicherplatz verbraucht haben, da die neue „TurtleGraphics" umfangreicher als die alte ist. Für Fenstertechnik ist 128K-Pascal + Maus erforderlich.

Im einzelnen bietet das Paket folgende Möglichkeiten:

- volle Kompatibilität mit der alten „TurtleGraphics"
- lauffähig auf Pascal 1.1 und 1.2
- funktionsfähig mit angeschlossener UltraTerm-Karte
- alle zeitkritischen Funktionen in reinem Assembler programmiert
- Benutzung der zweiten Hires-Seite ($4000–$5FFF) möglich
- für Apple IIc und Apple IIe mit erweiterter 80-Zeichen-Karte Benutzung der doppelten Hires-Grafik mit 560 × 192 Punkten bzw. 16 neuen Farben möglich
- schnelle Prozeduren zum Zeichnen eines Punktes oder einer Linie
- Linearisierung von Teilen des Hires-Schirms
- Benutzung mehrerer Zeichensätze gleichzeitig
- Laden und Speichern von Hires-Bildern mit Ausdruck über Pascal-SUPERDUMP
- Scrolling des Hires-Schirms oder eines Teils in vier Richtungen
- drei verschiedene Schriftarten: Fett-, Breit- und Proportionalschrift, beliebig mischbar (acht Möglichkeiten)
- spezielle schnelle Ausgabe von Text
- Cursor bei Eingabe frei programmierbar
- Ein-/Ausgabe von INTEGER-, CHAR-, STRING- und REAL-Werten im Grafikmodus
- Menüzeile wie beim Macintosh (Die nachfolgenden Module benötigen Maus/Joystick) und 128K-Pascal
- Pull-down-Menüs
- Laden und Speichern von Fenstern (Windows)
- Öffnen von Fenstern
- Aktivieren und Deaktivieren von Fenstern
- Verschieben und Vergrößern/Verkleinern von Fenstern
- Scrolling von Fensterinhalten in allen vier Richtungen
- Umfangreiche Demos als Quelltexte.

Hüthig Software Service · Postfach 10 28 69 · 6900 Heidelberg



EDITORIAL

Neue Preise ab 1. 2. 1987

Normale Peeker-Sammeldisketten

#1, #2, #3*, #4, #5*, #6, #7, #9*, #10, #11, #12, #13, #14, #15, #17, #18, #19, #21, #22, #23, #24, #25, #27
je Diskette DM 22,–
* vergriffen

Spezielle Peeker-Sammeldisketten

#8 mit Diversi-DOS,	DM 26,–
#16 mit Macroeditor,	DM 26,–
#20 mit Pic-Edit,	DM 26,–
#26 mit Dossembler,	DM 26,–

Software mit Handbüchern

Fast-Writer	
ProDOS-Version,	DM 98,–
DOS-3.3-Version,	DM 98,–
Turtle-Graphics-Paket,	DM 98,–
Superquick,	DM 38,–
DISK40,	DM 38,–
INPUT 2.0,	DM 38,–
MMU 2.0,	DM 38,–
ProDOS-Editor,	DM 38,–
Double-Hires-Tools	
Applesoft-Version,	DM 28,–
Kyan-Pascal-Version,	DM 28,–

vergriffen

Softbreaker
Superplot
Turtle-Graphics-Quelltexte
Pic-Edit-Quelltexte

Kyan-Pascal mit Tools

keine Preisänderungen, jedoch nur noch lieferbar, solange Vorrat reicht

Insgesamt erschienen seit September 1984 bis heute 28 Peeker-Ausgaben mit einem redaktionellen Umfang von etwa 9 Millionen Zeichen. Von den Sammeldisketten kamen ebenfalls 28 Exemplare heraus, die als DOS-lose Datendisketten über 3,5 Megabytes an Programmen enthalten.
Vor Ihnen liegt nunmehr die letzte Ausgabe des Peekers. Ich darf dies zum Anlaß nehmen, um mich bei allen Lesern zu bedanken, die uns zweieinhalb Jahre lang die Treue gehalten haben. Mein Dank gilt ferner den Aufsatzverfassern für ihre kompetenten Beiträge sowie den Fachberatern für ihre Unterstützung in technischen Detailfragen.
Unser Ziel war es stets, Ihnen möglichst nützliche Programme zu möglichst niedrigen Preisen zur Verfügung zu stellen. So kosteten beispielsweise unsere Sammeldisketten nicht mehr als Public-Domain-Disketten, obwohl bei diesen im Gegensatz zu unseren Disketten keine Autorenhonorare anfallen.
Auch bei den Peeker-Aufsätzen haben wir uns stets bemüht, in die systematisch gegliederten Beiträge möglichst viele Sachinformationen hineinzupacken. Dabei haben wir das vielerorts anzutreffende Larifari hoffentlich weitgehend vermieden.
Im Laufe der zweieinhalb Jahre haben wir einige tausend Leserbriefe beantwortet, von denen aus Platzgründen nur ein Bruchteil im Peeker veröffentlicht werden konnte. Oft war Hilfe möglich, oft aber auch nicht. Hierzu hätte es eines technischen Gesprächspartners bei der Firma Apple bedurft, den wir leider nie gefunden haben. Es wäre jetzt müßig, über verpaßte Chancen nachzugrübeln. Denn nachdem sich zunächst die „Apple's" und jetzt der Peeker von der Firma Apple abgewandt haben und nachdem in Kürze die AUGE das Wort „Apple" aus der Vereinssatzung streichen wird, dürfte sich das Problem von selbst gelöst haben.
Auch beim Apple IIgs bewegt sich überhaupt nichts. Im Moment ist nicht einmal klar, ob und wo das Gerät erhältlich sein wird. In der Schweiz jedenfalls soll der IIgs nicht in den Verkauf gelangen. Bei dieser Situation ist es nur natürlich, daß sich kein Verlag entschließen konnte, ein IIgs-Buch zu produzieren. Wenn Leute wie Steve Wozniak das Sagen gehabt hätten, wäre sicherlich ein anderes und besseres Gerät entwickelt worden.
Den Peeker-Lesern, die inzwischen auf Atari umgeschwenkt sind, empfehlen wir unser Diskettenmagazin, bei dem es sich übrigens nicht um Public-Domain-, sondern um Originalprogramme handelt, die auf den Disketten genau dokumentiert sind. Darüber hinaus verweisen wir die elektronisch-elektrotechnischen Fachleute unter unseren Lesern auf unsere einschlägigen Zeitschriften, z. B. „elektronik industrie", „productronic: Elektronik-Fertigung und Test", „industrie-elektrik + elektronik", „mini micro magazin", „CAE-Journal", „Intelligent Instruments & Computers", „nachrichten elektronik + telematik" u.a.m.

Ulrich Stiehl

inhalt

Assembler

65816/65802-Kompaktkurs
für Apple IIgs, Teil 2
von Arne Schäpers 6

Grafik

Double-Hires-Grafik-Paket
von Lothar Maier 14

Hardware

MultiRAM-CX-Karte
16-Bit-65C816 für den Apple IIc
von Lothar J. Dudek 22

Recht

Rechtschutz von Computerprogrammen
von Dipl.-Ing. Hans Raible 25

Telekommunikation

Videotext mit dem Apple II
von Dieter Charchot 30

Praxis

Bedienungsanleitung zum Dateiprogramm DB-Meister
von Ulrich Stiehl 31

Technik

Logik-Analysator
von Dipl.-Ing. Gerhard Berg 36

Impressum
Peeker
Magazin für Mikrocomputer
4. Jahrgang 1987
ISSN 0176–9200

© für den gesamten Inhalt einschließlich der Programme Dr. Alfred Hüthig Verlag, Heidelberg 1987

Verleger und Herausgeber:
Dipl.-Kfm. Holger Hüthig
Geschäftsführung Zeitschriften:
Heinz Melcher
Chefredakteur: Ulrich Stiehl (us)
Redaktion: Dagmar Berberich
Anzeigenleitung: Karl M. Dietzow
Anzeigendisposition: Diana Walter

Telefonnummern:
Zentrale: 06221/489-0
Redaktion: 06221/489-352
Anzeigen: 06221/489-206
Abonnement: 06221/489-283
Software: 06221/489-231
Bücher: 06221/489-353
(Bestellungen bitte nur schriftlich)

Abonnement:
Der Abonnent kann seine Bestellung innerhalb von 7 Tagen schriftlich durch Mitteilung an den Dr. Alfred Hüthig Verlag GmbH, Postfach 102869, 6900 Heidelberg 1, widerrufen. Zur Fristwahrung genügt die rechtzeitige Absendung des Widerrufs (Datum des Poststempels). Das Abonnement verlängert sich zu den jeweils gültigen Bedingungen um ein Jahr, wenn es nicht zwei Monate vor Jahresende schriftlich gekündigt wird. Die Abonnementsgelder werden jährlich im voraus in Rechnung gestellt, wobei bei Teilnahme am Lastschriftabbuchungsverfahren über die Postscheckämter und Bankinstitute eine vierteljährliche Abbuchung möglich ist.
Nichterscheinen infolge höherer Gewalt berechtigt nicht zu Ansprüchen gegen den Verlag.

Hüthig PUBLIKATION

peeker

Heft 3/1987

Testberichte

Arztec-C-Compiler
Ein Erfahrungsbericht
von Dipl.-Ing. K.-W. Bott 38

Print Shop
Ein Programmpaket für Matrixdrucker
von Dieter Charchot 42

The Print Shop Companion
Die Print-Shop-Erweiterung
getestet von Hans-Martin Eng 43

EXTRA K
Anwendungen für die erweiterte
80-Zeichenkarte
von Franz-Josef Hüskens 44

Visible Computer
Ein Assembler zum Zuschauen
von Dagmar Berberich 48

Deutsch lernen mit dem Apple
Ein Computerprogramm
zur Übung der Präpositionen 49

CADAPPLE
Ein CAD-System für den Apple II
getestet von Hans-Martin Eng 50

Neue Firmware für Ehring-V80-Karte
von Dagmar Berberich 52

Atari

MUSIX32
Ein Musik-Editor für den Atari ST
getestet von Dieter Geiß 53

Atari-Produkte
ST Pascal Plus u. a. 55

Das Simulationsprogramm Umweltdynamik
von Dieter Charchot 56

Literaturverwaltungsprogramm Bookends
von Dieter Charchot 57

Schreiben mit der Maus
Textverarbeitungsprogramm Multiscribe
getestet von Reinhard Frank 59

Macroworks
Eine Erweiterung für Appleworks
von Franz-Josef Hüskens 60

Die Soundkarte ADD 6
getestet von Hans-Martin Eng 61

Bücher 62

Leserbriefe 65

Register
Peeker-Sammeldisketten
Gesamtverzeichnis #1 bis #27 67

Anschrift:
Dr. Alfred Hüthig Verlag GmbH
Im Weiher 10, Postfach 10 28 69
6900 Heidelberg
Telefon (0 62 21) 4 89-0
Telex 4-6 17 27 hued d.
Telefax (0 62 21) 489 279
BTX * 51851 #

Auslieferung für die Schweiz:
Delta-Verlag
Herr R. de Forest
Gugelmattstraße 31
8967 Widen
Telefon 057 / 33 86 86

Vertrieb:
Erscheinungsweise: 12 Hefte jährlich,
Erscheinungstag jeweils 1 Woche vor Monatsbeginn.
Jahresabonnement Inland DM 75,–, einschl. MwSt und Versandkosten.
Jahresabonnement Ausland DM 75,– plus DM 20,– Versandkosten.
Einzelheft DM 6,50
Vertrieb Handel:
Vertriebsleitung:
Walter Menzel, Tel. (0 62 21) 48 92 80

Bankverbindungen:
Zahlungen: an den Dr. Alfred Hüthig Verlag GmbH, D-6900 Heidelberg 1: Postgirokonten: Ludwigshafen 4799-673, BLZ 545 100 67; Österreich: Wien 75558 88; Schweiz: Basel 40-24417-4; Niederlande: Den Haag 1 457 28; Italien: Mailand 5 968 92 08; Belgien: Brüssel 07 230 26-85; Dänemark: Kopenhagen 603 4969; Norwegen: Oslo 199 4243; Schweden: Stockholm 5477 76-5
Bankkonten: Landeszentralbank Heidelberg 67 207 341; BLZ 672 000 00; Deutsche Bank Heidelberg 02 65 041; BLZ 672 700 03; Bezirkssparkasse Heidelberg 204 51, BLZ 672 500 20.

Herstellung:
Produktionsleitung: Gunter Sokollek
Gestaltung: Rainer Schmitt
Titelbild: Werner Hable
Satz und Druck:
Heidelberger Verlagsanstalt
Printed in Germany

65816/65802-Kompaktkurs für Apple IIgs

Teil 2

von Arne Schäpers

6. Adressierungsarten

Die folgende Unterteilung zwischen „alt" und „neu" dürfte den Lesern, die nicht an einen 65(C)02 gewöhnt sind, etwas ungewöhnlich vorkommen. Da ich aber davon ausgehe, daß sich der weitaus größere Teil der Peeker-Leser primär mit diesem Prozessor beschäftigt, erscheint sie mir gerechtfertigt.

6.1. Adressierungsarten des 65(C)02

Der 65816 kennt sowohl im „Emulation Mode" als auch im „Native Mode" sämtliche Adressierungsarten des 65(C)02. Achtung: Der 65802 addiert zu sämtlichen „Nullseitenadressen" den Inhalt des D-Registers hinzu, der 65816 stellt „absoluten" Adressen zusätzlich das Datenbank-Register (DB) voran – und zwar sowohl im „Native Mode" als auch im „Emulation Mode".

6.1.1. Implied (implizit) – i

Unter diesem Begriff werden sämtliche Befehle zusammengefaßt, bei denen sowohl die Quelle als auch das Ziel des Transfers durch den Befehl selbst vorgegeben sind, d.h. Adreßangaben fehlen. Das ist für alle Register-zu-Register-Operationen der Fall: Quell- und Zielregister sind durch den Befehl selbst festgelegt. Befehle mit impliziter Adressierung sind nur 1 Byte lang. Beispiele:

```
TXA          ; Inhalt von X -> Akku
TCD          ; Inhalt von A -> Direct
DEX          ; Inhalt von X minus 1 -> X
```

Zwischen dem 65(C)02 und dem 65816 besteht hier kein Unterschied – wenn man davon absieht, daß der letztere über eine Reihe neuer Transferbefehle verfügt, mit denen die zusätzlichen Register erfaßt werden. Ein Befehl wie PHA ist *nicht* implizit, obwohl auch hier Quelle und Ziel durch den Befehl selbst festgelegt ist. Der Grund: Die tatsächlich angesprochene Adresse hängt vom Stand des Stackpointers ab.

6.1.2. Immediate (unmittelbar) – #

Hier folgt der Wert des Operanden selbst unmittelbar auf den Befehlscode (Kennzeichen „#" vor dem Wert des Operanden). Bedingt durch die umschaltbare Registerbreite des 65816 kann es sich dabei entweder um ein oder um zwei Bytes handeln. Beispiele:

```
LDA #$93        ; Akku <- Wert $93
AND #$7F        ; AND mit Wert $7F
CMP #$2Ø        ; Vergleich mit Wert $2Ø
LDX #$3849      ; X-Register <- $3849
LDX #$12        ; X-Register <- $12
LDY #$17        ; Y-Register <- $17
```

Die unterschiedliche Wortbreite des 65816 wirft hier ein großes Problem auf – der Befehlscode („Opcode") für 8- und 16-Bit-Operationen ist derselbe. Ob der Prozessor ein oder zwei Bytes liest, hängt *nur vom Zustand der Flags M und X* ab. Wenn sich die drei letzten im Beispiel benutzten Befehle tatsächlich in der gezeigten Reihenfolge befinden würden, wäre ihre Interpretation durch den Prozessor folgendermaßen:

```
; Indexregister mit 8 Bit Breite
LDX #$49        ; X-Reg <- Wert $49
SEC             ; $38 als Befehl!
LDX #$12        ; wird wieder "korrekt"
LDY #$17        ; interpretiert

; Indexregister mit 16 Bit Breite
LDX #$3849      ; X-Reg <- Wert $3849
LDX #$AØ12      ; "LDY" als höherw. Byte
ORA [..],y ; $17 als Befehl!
```

Dieses Problem ist unlösbar – sämtliche mir bekannten Assembler/Disassembler benötigen deshalb eine explizite Angabe, für welche Registerbreite Befehle erzeugt bzw. interpretiert werden sollen.

6.1.3. Akkumulator – A

Diese Adressierungsart wird von Operationen benutzt, bei denen der Akkumulator sowohl die Quelle als auch das Ziel darstellt (eigentlich fällt das unter den Begriff „implizit").
Alle sechs Befehle, die mit dieser Adressierungsart arbeiten, sind lediglich ein Byte lang: Schiebe- und Rotationsbefehle, die nicht auf eine Speicheradresse, sondern auf den Akku-Inhalt angewendet werden (ASL, LSR, ROL und ROR – vgl. 3.6.7.) und die (erst ab dem 65C02 implementierten) Operationen INA und DEA (Akku-Inhalt plus 1/minus 1).
Abgesehen von der Tatsache, daß die Breite des Akkus beim 65816 zwischen 8 und 16 Bits umgeschaltet werden kann (im zweiten Fall operieren alle sechs Befehle mit 16-Bit-Worten), gibt es keine Unterschiede zwischen 65(C)02 und 65816.

6.1.4. Direct – d
(Seite 0 oder „Zp" = Zeropage)

Hier folgt kein Wert, sondern eine Adresse auf den Befehlscode – der Prozessor arbeitet mit dem Inhalt der dadurch bezeichneten Speicherstelle. Die Adresse wird immer nur mit *einem* Byte angegeben, die adressierte Speicherstelle befindet sich auf der Seite 0:

```
LDA $AØ         ; Akku <- Inhalt v.$ØØAØ
STA $BØ         ; $ØØBØ <- Akku-Inhalt
CPY $Ø3         ; Vergleich Y-Reg – $ØØØ3
```

Bei den meisten anderen Prozessoren wird durch eine besondere Kennzeichnung zwischen „Wert" und „Speicheradresse" unterschieden, etwa durch runde oder eckige Klammern (Z80: LD A,(00A0), 8086: MOV AL,[00A0]) oder durch einen vorangestellten „Klammeraffen" wie bei der 80x2-Serie und einigen 68000-Assemblern.

Zwischen dem 65(C)02 und dem 658xx gibt es zwei Unterschiede:
– Wenn das beteiligte Prozessorregister auf 16 Bit Breite gesetzt ist, werden zwei aufeinanderfolgende Speicherstellen gelesen. Ein Befehl wie LDX $A0 liest also die Speicherstellen $00A0 und $00A1, wobei $00A0 den niederwertigen Teil enthält.
– Zu der angegebenen Adresse wird grundsätzlich (d.h. auch im „Emulation Mode") der Inhalt des D-Registers dazuaddiert. Die gezeigten Beispiele gelten also nur für D = $0000. Hat D den Wert $0100, dann liest der Befehl LDA $A0 die Speicherstelle $00/01A0 (und evtl. $01A1), für D = $0200 wird $00/02A0 gelesen usw.
„Nullseitenadressierungen" sprechen grundsätzlich Speicherstellen innerhalb der Bank 00 an (s.a. Abschnitt 3.3.).

6.1.5. Absolute – a oder „ABS"

Diese Adressierungsart ist in den Beispielen der vorangegangenen Abschnitte bereits häufig verwendet worden. Auf den Befehlscode folgen zwei Bytes, die als 16-Bit-Adresse interpretiert werden, der Prozessor arbeitet mit dem Inhalt der dadurch bezeichneten Speicherstelle. Beispiele:

```
LDA $A000       ; Akku <- Inhalt v. $A000
STX $4387       ; $4387 <- Inhalt von X
CMP $1700       ; Vergleich Akku - $1700
```

Abgesehen von der Tatsache, daß eine 16-Bit-Operation des 65816 zwei aufeinanderfolgende Adressen liest, gibt es einen weiteren grundsätzlichen Unterschied: Einer absoluten Adresse wird beim 65816 immer der Inhalt des Datenbank-Registers (DB) vorangestellt. Wenn DB den Inhalt $00 hat, dann bezieht sich der Befehl LDA $A000 auf die Adresse $00/A000, für DB = $01 wird $01/A000 angesprochen, für DB = $7E die Adresse $7E/A000 usw.
Die Bezeichnung „absolut" wird verwirrenderweise auch für absolute Sprünge (JMP) und Unterprogrammaufrufe (JSR) verwendet, wobei diese Definition gleich in doppelter Hinsicht widersprüchlich ist. Zum einen folgt das Sprungziel unmittelbar auf den Befehl im Programmcode (ein Befehl wie JMP $2314 wird als $4C $14 $23 kodiert) und ist damit eigentlich „immediate", zum anderen wird hier natürlich nicht das Datenbank-Register, sondern das Programmbank-Register vorangestellt.

6.1.6. Absolute Indexed (absolut indiziert) – a,X / a,Y oder ABS,X / ABS,Y

Auf den Befehlscode folgt eine absolute „Basisadresse", die aus zwei Bytes besteht. Zu dieser Basis wird der Inhalt eines Indexregisters hinzuaddiert. Das Ergebnis dieser Addition ist die Adresse der Speicherstelle, mit deren Inhalt der Prozessor arbeitet. Beispiele:

```
LDX #$00        ; X-Reg <- Wert $00
LDA $2000,X     ; Akku <- Inh. von $2000
INX             ; X von $00 auf $01
LDA $2000,X     ; Akku <- $2001
LDY #$20        ; Y-Reg <- Wert $20
AND $2000,Y     ; AND mit Inh. von $2020
STA $6534,Y     ; $6554 <- Akku-Inhalt
```

Die Indexregister des 65(C)02 sind grundsätzlich 8 Bit breit, können also Werte im Bereich von $00..$FF annehmen. Da der 65(C)02 maximal 64K adressieren kann, führt eine Befehlsfolge wie

```
LDX #$20
LDA $FFF0,X     ; $FFF0+$20 = $10010
```

zur Adressierung der Speicherstelle $0010 – dasselbe ist beim 65802 der Fall. Zum 658xx bestehen die folgenden Unterschiede:

– Jeder Adressierung wird der Inhalt des Datenbank-Registers (DB) vorangestellt. Wenn DB den Wert $00 hat, dann adressiert der Befehl LDA $2000,X die Speicherstelle $00/2000,X; hat DB den Wert $01, wird $01/2000,X angesprochen usw.
– Abhängig von der Breite des Akkus werden eine oder zwei Speicherstellen adressiert. Der Befehl LDA $2000,X liest also die Speicherstellen DB/2000,X und DB/2000,X plus 1 (etwas einfacher: DB/2001,X), wenn das M-Flag gelöscht ist.
– Die Indexregister können von 8 auf 16 Bits geschaltet werden. Der mit einem Befehl wie LDA $2000,X erreichbare Bereich erweitert sich in diesem Fall von DB/2000..DB/20FF auf DB/2000..DB+1/1FFF.

6.1.7. Direct Indexed (direkt indiziert) – d,X /d,Y oder Zp,X / Zp,Y

Diese Adressierungsart stellt eine Kombination von „direct" und „absolute indexed" dar: Auf den Befehlscode folgt eine Adresse, die mit einem Byte angegeben wird. Der Inhalt des Indexregisters wird zu dieser Adresse dazuaddiert, das Ergebnis ist die Adresse der Speicherstelle, mit deren Inhalt der Prozessor arbeitet. Beispiele:

```
LDX #$14        ; X-Reg <- Wert $14
LDA $12,X       ; Akku <- Inh. von $0026
STA $FF,X       ; ???
```

Der dritte Befehl dieses Beispiels soll den ersten Unterschied zwischen dem 65(C)02 und dem 658xx verdeutlichen: Im „Native Mode" wird der Inhalt des Akkus in die Speicherstelle $0113 (= $FF + $14) geschrieben – der 65(C)02 bzw. der „Emulation Mode" berücksichtigt den Übertrag über die Seite 0 hinaus nicht und adressiert die Speicherstelle $0013.
Im „Native Mode" werden auch Überträge berücksichtigt, die durch 16 Bit breite Indexregister hervorgerufen werden. Die Befehlsfolge

```
LDX #$6000      ; 16 Bits: X <- $6000
STA $50,X
```

schreibt tatsächlich den Inhalt des Akkus in die Speicherstelle $6050.
Der wichtigste Unterschied zwischen 65(C)02 und 65816 liegt in der Verwendung des D-Registers. Als wenn es noch nicht kompliziert genug wäre, wird der aus „Basis plus Indexregister" gewonnenen Adresse zusätzlich der Inhalt von D hinzuaddiert. Wenn D den Inhalt $0000 hat, dann adressiert das zuvor gezeigte Beispiel die Speicherstelle $00/6050; hat D den Inhalt $0100, dann wird $00/6150 angesprochen usw.
Wie in 3.3. bereits erwähnt, läßt sich die Bank 00 allerdings nicht überschreiten: Wenn D den Inhalt $F000 hat, dann wird für das gezeigte Beispiel nicht etwa die Speicherstelle $01/5050 adressiert ($F000 plus $50 plus $6000 = $015050), sondern $00/5050 – die Adressierung bewegt sich also innerhalb der Bank $00 im Kreis.
Der Vollständigkeit halber: Wenn der Akku auf 16 Bit Breite gesetzt ist, dann werden zwei aufeinanderfolgende Speicherstellen angesprochen, wobei der Inhalt der ersten als niederwertiger Teil interpretiert wird.

6.1.8 Direct Indirect Indexed oder Indirekt,Y – (d),Y oder (Zp),Y

Bei dieser Adressierungsart folgt eine „Nullseitenadresse" (ein Byte) auf den Befehlscode. Der Inhalt dieser und der darauffolgenden Speicherstelle wird gelesen und als 16-Bit-Wort interpretiert. Zu diesem Wort wird der Inhalt des Y-Registers dazuaddiert. Das Ergebnis ist die Adresse der Speicherstelle, mit deren Inhalt der Prozessor arbeitet. Die beiden „Nullseitenadressen" zeigen also auf eine Basisadresse, sie werden deshalb auch *Pointer* (= Zeiger) genannt. Beispiel:

```
LDA #$2000      ; Akku <- Wert $2000
STA $15         ; $15/$16 als Zeiger
LDY #$0000
LDA ($15),Y     ; lädt $2000
INY             ; Y-Reg. von 0 auf 1
LDA ($15),Y     ; lädt $2001
```

Äquivalentes ließe sich über die Adressierungsart „ABS,Y" erreichen (s. 6.1.5.):

```
LDY #$0000
LDA $2000,Y     ; lädt $2000
INY
LDA $2000,Y     ; lädt $2001
```

Der Vorteil gegenüber der absoluten Indizierung liegt darin, daß für eine Veränderung der Basisadresse lediglich die entsprechenden Speicherstellen auf der Seite 0 verändert werden müssen – und nicht der Programmcode selbst. Um dieselben Befehle auf einen Datenbereich anzuwenden, der z.B. mit der Adresse $4000 beginnt, muß im ersten Fall lediglich der „Pointer" $15 verändert werden – im zweiten Fall müßte man sämtliche „LDA $2000,Y"-Befehle durch „LDA $4000,Y" ersetzen.

Abgesehen von der Möglichkeit, die Register auf 16 Bit Breite zu schalten, kommen beim 65816 zwei Erweiterungen hinzu, die den gesamten Prozeß leider noch komplizierter machen, als er schon ist:

– Der Zugriff auf den „Pointer" ist eine Adressierung der „Seite 0", folglich wird der Inhalt des D-Registers dazuaddiert. Der Befehl LDA ($15),Y liest also die Speicherstellen $00/0015 und $00/0016, wenn D den Wert $0000 hat; für D = $0100 werden $00/0115 und $00/0116 gelesen usw.
– der Zugriff auf die letztendlich adressierte Speicherstelle ist „absolut", ihr wird der Inhalt des Datenbank-Registers vorangestellt. Für das oben gezeigte Beispiel ergibt sich die Adressierung der Speicherstelle $00/2000, wenn DB den Wert $00 hat; für DB = $01 wird $01/2000 gelesen usw.

6.1.9. Direct Indexed Indirect – (d,X) oder (Zp,X)

Hier folgt ebenfalls eine „Nullseitenadresse" auf den Befehlscode. Zu ihr wird der Inhalt des X-Registers hinzuaddiert. Das Ergebnis dieser Addition ergibt eine „Pointeradresse": Der von dort gelesene Wert ist die Adresse der Speicherstelle, mit deren Inhalt der Prozessor arbeitet. Die Entwickler der 65er-Familie haben sich diese Adressierungsart zur Verwaltung von Zeigertabellen ausgedacht. Aus zwei Gründen wurde sie jedoch bisher nur äußerst selten benutzt:
1) Der 65(C)02 verfügt nur über eine einzige Zeropage – und die ist bei allen leistungsfähigeren Systemen so übervoll, daß Pointertabellen in den „normalen" Speicherbereich ausgelagert werden müssen.

2) Diese Adressierungsart ist so kompliziert, daß sie nur von sehr wenigen Programmierern überhaupt verstanden wird.

Beim 65816 entfällt zumindest der erste Grund. Ein Beispiel für die Verwendung von „(Zp,X)": Gegeben seien mehrere Datenstrukturen unterschiedlicher Größe, deren Startadressen auf der Seite 0 verzeichnet sind, z.B.:

```
ØØ6Ø: ØØ 12 24 12 96 12 14 13
```

Die Speicherstellen $0060/61 enthalten einen Zeiger auf die Adresse $1200; $0062/63 zeigen auf die Adresse $1224; $0064/65 auf $1296 usw.
Jede Datenstruktur enthalte auf ihrer Startadresse wiederum einen Pointer. Apple, Inc. bezeichnet ein derartiges Konstrukt (einen „Pointerpointer") in der Dokumentation des IIgs als „Handle".
Eine Routine, die ein „dereferencing" vornimmt, d.h. den aktuellen Wert eines „Handle" liest, gestaltet sich unter Zuhilfenahme der Adressierungsart „(Zp,X)" mehr als einfach – im gezeigten Beispiel geschieht der Prozeß über die Nummer der gewünschten Datenstruktur im Akku:

```
ASL             ; Akku mal 2
TAX             ; X-Reg <- Akku-Inhalt
LDA ($6Ø,X)     ; (16 Bits!)
STA $5Ø         ; Pointer auf die end-
RTS             ; gültige Adresse
```

Wenn der Akku beim Aufruf den Inhalt 0 hat, bekommt X ebenfalls den Inhalt 0, der Befehl LDA ($60,X) lädt den Inhalt der Speicherstellen $1200/01. Für X = $02 wird der Inhalt der Speicherstellen $1224/25 geladen, für X = $04 liest der Prozessor die Adressen $1296/97 usw.

Der 65816 verwendet in diesem Fall das D-Register in doppelter Weise, nämlich sowohl für die Basisadresse ($60) als auch für die Adresse des ersten Pointers. Wenn D den Inhalt $0000 hat und das X-Register auf $0002 gesetzt ist, dann liest der Prozessor in unserem Beispiel die Adressen $00/0062 und $00/0063 als Pointer; für D = $0100 werden $00/0162 und $00/0163 gelesen.

Durch die Tatsache, daß das X-Register auch mit 16 Bit Breite arbeiten kann, sind theoretisch bis zu 64K große „Handle"-Tabellen möglich. Ein Adressierungsversuch über die Adresse $00/FFFF hinaus führt wie bei allen „Nullseitenadressen" wieder zu einer Adresse in der Bank 00 – ein Überlauf in die Bank $01 hinein findet nicht statt.
Der Adressierung der Speicherstelle, deren Inhalt letztendlich gelesen wird, ist das Datenbank-Register (DB) vorangestellt. Im gegebenen Beispiel werden $00/1200 und $00/1201 gelesen, wenn DB den Inhalt $00 hat; für DB = $01 ergibt sich $01/1200 und $01/1201 usw.

6.1.10. Program Counter Relative (relativ zum PC) – r

Diese Adressierungsart wird beim 6502 nur auf den Programmzähler (PC) und im Zusammenhang mit bedingten Sprüngen angewendet. Auf den Befehlscode folgt ein Byte „Sprungoffset", das als vorzeichenbehaftet interpretiert wird: Ein Wert im Bereich von $00..$7F führt zu einem Sprung in Richtung höherer Adressen, ein Wert im Bereich von $80..$8F zu einem Sprung in Richtung niedrigerer Adressen. Durch diese Vereinbarung haben relative Sprünge eine maximale Reichweite von -126 bis +127 Byte.
Ein Beispiel zur Absuche eines Speicherbereiches von $37 Byte Länge nach dem Zeichen „*":

```
      LDX #$ØØ
LOOP  LDA $2ØØØ,X        ; $2ØØØ, $2ØØ1...
      CMP #'*'           ; ist "*"?
      BEQ FOUND          ; wenn ja: Sprung
      INX                ; X-Reg plus 1
      CPX #$37           ; Ende erreicht?
      BCC LOOP
      <...>              ; nicht gefunden!

FOUND <...>              ; gefunden
```

Die absolute Adresse dieses Programmfragments spielt keine Rolle, weil sämtliche darin vorkommenden Verzweigungen *relativ* zum momentanen Stand des PC stattfinden. Der erste Sprung („BEQ") hat ein positives Argument, der zweite („BCC") ein negatives.
Zusätzlich zu den bedingten Sprüngen kennt bereits der 65C02 einen „unbedingten" relativen Sprung (BRA – „Branch Relative Always"), der auf dieselbe Weise arbeitet – nur daß seine Ausführung nicht von dem Zustand eines Flags abhängig gemacht wird.
Der 65816 wurde in diesem Punkt lediglich um einen einzigen Befehl erweitert: BRL („Branch Relative Long") arbeitet nicht mit einem, sondern mit zwei Byte „Sprungoffset" und ist genauso wie BRA nicht von einer bestimmten Bedingung abhängig. Mit zwei Bytes lassen sich Werte von 0 bis 65535 ausdrücken, die Reichweite dieses Befehls geht also von -32766 bis +32767.
Der Wechsel der Programmbank ist über relative Sprünge grundsätzlich *nicht* möglich.

6.1.11. Absolute Indirect (absolut indirekt) – (a) oder (ABS)

Diese Adressierungsart wird ebenfalls nur auf den Programmzähler (PC) angewendet. Auf den Befehlscode folgt ein 16-Bit-Wort (2 Bytes), das der Prozessor als Adresse interpretiert. Ab dieser Adresse werden zwei aufeinanderfolgende Speicherstellen gelesen, mit ihrem Inhalt wird der PC neu gesetzt. Ein (altbekanntes) Beispiel: Gegeben seien die Speicherstellen

```
ØØ36- FØ FD 1B FD
```

Der Befehl JMP ($0036) führt damit zu einem Sprung zur Adresse $FDF0, nach einem JMP ($0038) setzt der Prozessor das Programm mit der Adresse $FD1B fort. Daß dieses Beispiel Adressen im Bereich $00xx verwendet, ist reiner Zufall – ein „absolute indirect direct" ist nicht definiert.

Zwischen 65(C)02 und 658xx bestehen nur die „üblichen" Unterschiede:
– Das Argument des JMP-Befehls wird als „Daten" betrachtet. Wenn DB den Wert $00 hat, dann liest der Befehl den Inhalt der Speicherstellen $00/0036 und $00/0037 als Vektor, für DB = $01 werden $01/0036 und $01/0037 gelesen usw.
– Der Inhalt von PB wird durch diesen Befehl nicht verändert. Ein indirekter Sprung findet grundsätzlich nur innerhalb der momentan gesetzten Programmbank statt.

6.1.12. Stack – s

Diese Adressierungsart ist beim 65(C)02 die einzige, die den Stackpointer (SP) verwendet, und ist zudem recht einfach ausgefallen. Unter ihr werden alle Befehle zusammengefaßt, die etwas auf dem Stack speichern oder von dort wieder lesen. Das folgende Beispiel tauscht die Register A und X aus und benutzt dazu den Stack:

```
PHA             ; Akku auf den Stack
PHX             ; X auf den Stack
PLA             ; Akku <- X-Inhalt
PLX             ; X-Reg <- Akku-Inhalt
```

Unterprogrammaufrufe (JSR) und -rücksprünge (RTS) werden ebenfalls dieser Adressierungsart zugeordnet. Ein JSR speichert zuerst eine Returnadresse auf dem Stack, bevor der angegebene Wert in den PC eingesetzt wird. Beispiel: Wenn der Prozessor von der Adresse $1730 den Befehl

```
173Ø-     JSR $1812
1733-     ...
```

liest, wird zuerst der Stand des PC auf den Stack gebracht und dann ein Sprung zur Adresse $1812 ausgeführt. Innerhalb dieses Unterprogramms hat der Stack den folgenden Inhalt:

```
      $17    PC, höherw. Teil
      $32    PC, niederw. Teil
SP  ->
```

Auffallen sollte hier, daß der gespeicherte Wert um eins zu niedrig ist – im „Hauptprogramm" geht es schließlich mit der Adresse $1733 weiter. Dieses Verhalten ist auf das erste Design des 6502 zurückzuführen und mußte aus Kompatibilitätsgründen für alle weiteren Mitglieder der 65er-Familie beibehalten werden. Die „Korrektur" erfolgt erst durch den RTS-Befehl, der eine Erhöhung des vom Stack gelesenen Wertes um $0001 vornimmt, bevor das Ergebnis wieder in den PC eingesetzt wird (vgl. 5.2.1).
Abgesehen von der Tatsache, daß der Stackpointer des 65816 im „Native Mode" eine Breite von 16 Bits hat und je nach Breite des „gepushten" bzw. „gepullten" Registers um eins oder zwei erniedrigt wird (das zuerst gezeigte Beispiel funktioniert nur, wenn A und X dieselbe Breite haben!), existiert noch eine Reihe zusätzlicher Befehle, mit denen die neuen Register des 658xx (d.h PB, DB und D) auf den Stack gebracht bzw. wieder davon heruntergeholt werden können.

6.2 Neue Adressierungsarten des 65816

Der 65816 wurde gegenüber dem 65(C)02 um insgesamt 10 Adressierungsarten erweitert. Die Erweiterungen lassen sich in die folgenden Klassen einteilen:
● Vereinfachungen

- Behandlung „langer" Adressen und Pointer, d.h. Adressierungsverfahren, die mit 24 Bits arbeiten
- Wesentlich erweiterte Operationen über den Stack
- Erweiterungen für Sprünge und Unterprogrammaufrufe
- Spezialbefehle zum Transfer von Datenblöcken

6.2.1. Direct Indirect – (d) oder (Zp)

Diese Adressierungsart benutzt einen Pointer auf der Seite 0 und arbeitet in derselben Weise wie „(Zp),Y", nur daß hier der Inhalt des Y-Registers ignoriert wird. Beispiel: Wenn die Speicherstellen $15/$16 auf der (über das D-Register gesetzten) Seite 0 den Inhalt $28 und $40 haben, also auf die Adresse DB/$4028 zeigen, dann läßt sich die Befehlsfolge

```
LDY #$00
LDA ($15),Y     ; lädt DB/$4028
```

durch die Adressierungsart „(Zp)" folgendermaßen vereinfachen:

```
LDA ($15)       ; lädt DB/$4028
```

Da die Addition des Y-Registers auch prozessorintern entfällt, benötigt diese Operation außerdem einen Taktzyklus weniger als „(Zp),Y".

6.2.2. „Absolut lange" Adressen – al / al,X / al,Y

Diese drei Adressierungsarten stellen eine direkte Erweiterung der Modi „ABS", „ABS,X" und „ABS,Y" dar. Anstelle einer 16-Bit-Adresse folgt hier eine 24-Bit-Adresse auf den Befehlscode. (Um Irrtümern vorzubeugen: „absolut lange" Adressierungen haben grundsätzlich eigene Befehlscodes, die sie von „absoluten" Adressierungen unterscheiden. Probleme wie mit „kurzen" und „langen" Ladeoperationen von Registern gibt es hier nicht.)

Eine Operation über „al" ignoriert den Inhalt des DB-Registers, verändert wird dieses Register aber nicht. Die folgenden Beispiele gehen davon aus, daß das DB-Register den Wert $41 hat:

```
LDA $2040       ; DB/$2040: $412040
STA $24A4D5     ; DB ignoriert
LDA $2041       ; DB/$2041: $412041
STA $24A4D6     ; DB ignoriert

LDX #$2000      ; X-Reg <- Wert $2000
LDY #$3000      ; Y-Reg <- Wert $3000
LDA $000000,X   ; = $00/2000
STA $604011,Y   ; = $60/7011
STA $0800,Y     ; DB/3800: $413800
STA $20,X       ; ???
```

Die letzten vier Befehle dieses Beispiels verdeutlichen, wie unterschiedlich die Wirkung je nach Adressierungsart ausfallen kann: Ein Befehl wie „LDA $000000,X" benutzt zwar die Adresse $00/0000 als Basis, hat aber *nichts* mit der Seite 0 zu tun. Der Befehl „STA $0800,Y" benutzt eine „absolute" Adresse, arbeitet also mit einem 16-Bit-Operanden: Hier wird das DB-Register vorangestellt. Der Befehl „STA $20,X" hat nur ein Byte als Operand und ist eine „Nullseitenadressierung" – hier wird der momentane Inhalt des D-Registers zur Basisadresse $20 dazuaddiert, der Inhalt des Akkus landet auf der Adresse $00/ (D-Register plus $20 plus X-Register).

An dem Terminologiewirrwarr sind die Entwickler des 65816 unschuldig – natürlich wäre es einsichtiger, wenn eine „absolute" Adresse auch absolut (und nicht relativ) zu DB wäre, aber hier tritt eben wieder die Erblast des 65(C)02 zutage. (Was passiert, wenn eines Tages ein „651632" auf den Markt kommt, der mit 32-Bit-Adressen arbeitet? Dann kommt zu den Bezeichnungen „ABS" und „al" wahrscheinlich noch ein „ABB" (= really absolute, you'd better believe it) dazu...)

6.2.3. „Absolut lange Pointer" – [d] / [d],Y oder [Zp] / [Zp],Y

Diese beiden Adressierungsarten unterscheiden sich von „(Zp)" und „(Zp),Y" durch die Verwendung eines Pointers mit einer Länge von 24 Bits (d.h. drei Bytes). Das folgende Beispiel geht von einigen Voraussetzungen aus:

```
; D-Register = $0000
; $00/0015- 17 39 42
LDA [$15]       ; lädt $42/3917
LDY #$0000
LDA [$15],Y     ; dito
INY             ; Y von 0 auf 1
LDA [$15],Y     ; lädt $42/3918
```

Ein Befehl wie „LDA [$15]" liest also die Speicherstellen $00/0015, $00/0016 und $00/0017, interpretiert ihren Inhalt als „absolut lange" Adresse und lädt den Akku mit dem Inhalt der darüber angesprochenen Speicherstelle. Wenn D den Wert $0100 hat, dann werden die Speicherstellen $00/0115..$00/0117 als Pointer gelesen usw.

Der momentane Inhalt des DB-Registers wird für die Dauer der Operation ignoriert, verändert wird er aber nicht.

Da sich der 65816 im „full native mode" (E-Bit gelöscht, alle Register 16 Bits) mit einem 3-Byte-Wort ziemlich schwer tut, verwendet der Apple IIgs für „lange" Pointer grundsätzlich vier Bytes. Das höchstwertige Byte sollte dabei immer den Wert $00 haben. Das folgende Beispiel benutzt die Terminologie des ORCA/M-Assemblers und setzt PTR (4 Bytes auf der Seite 0) auf die „absolut lange" Adresse des Datenkonstrukts OBJECT:

```
LDA #OBJECT         ; untere 16 Bits
STA PTR             ; PTR auf Seite 0
LDA #OBJECT|-16     ; obere 16 Bits
STA PTR+2
```

Wenn OBJECT die (vom Assembler ermittelte) Speicheradresse $40/1729 hat, dann haben die Speicherstellen PTR..PTR+3 danach den folgenden Inhalt:

```
29 17 40 00
```

Ein Befehl wie „LDA [PTR]" lädt den Inhalt der Speicherstellen OBJECT+0 und OBJECT+1 in den Akku; die Befehlsfolge „LDY #$0020 / LDA [PTR],Y" lädt die Inhalte der Speicherstellen OBJECT+$0020/21 usw.

6.2.4. Sprünge und Unterprogrammaufrufe – (a) / (a,x) oder (ABS) / (ABS,X)

Die erste Form wurde bereits in 6.1.11. besprochen: Wenn die Speicherstellen $0036 und

Preiswerte Begleitdisketten

Bd. 1: DM 28,–; Bd. 2: DM 28,–

DM 28,–

DM 28,– (Neue Diskette für 3. Aufl.)

Hüthig Software Service
Postfach 10 28 69 · 6900 Heidelberg

$0037 den Inhalt $F0 und $FD haben, dann setzt der Befehl „JMP ($0036)" das Programm ab der Adresse $FDF0 fort. Der Inhalt des Programmbank-Registers (PB) wird dadurch nicht beeinflußt. Die Adressierungsart „(ABS)" kann aber zusätzlich für den Befehl JML („Jump Long") verwendet werden – hier werden nicht zwei, sondern drei Speicherstellen hintereinander gelesen, der Inhalt der dritten setzt das PB-Register neu. Beispiel: Wenn die Speicherstellen $40/248A, $40/248B und $40/248C den Inhalt $17, $12 und $38 haben, dann setzt der Befehl

```
JML ($248A)
```

die Programmausführung mit der Adresse $38/1217 fort. Ein Fallstrick: Die Adresse $248A ist „absolut", d.h. wird über den Inhalt von DB angesprochen. Das Beispiel funktioniert also nur in der gezeigten Form, wenn DB den Inhalt $40 hat.

Völlig neu und sehr erfreulich ist die zweite Adressierungsart für die Befehle JMP und JSR (nicht aber JML und JSL): Über den Inhalt des X-Registers wird eine Tabelle mit Startadressen ausgelesen („berechnetes GOTO"). Anstelle langer Erklärungen ein Beispiel: Gegeben sei ein Programm mit mehreren Funktionen, deren Startadressen folgendermaßen verzeichnet sind:

```
$4Ø/12ØØ- AB 17      ; Funktion Ø
$4Ø/12Ø2- Ø2 18      ; Funktion 1
$4Ø/12Ø4- 38 19      ; Funktion 2
```

Das folgende Programmfragment wird mit der Nummer der gewünschten Funktion (0, 1, 2...) im Akku aufgerufen, liest die Tabelle und führt einen entsprechenden Sprung (oder ein JSR) aus. Voraussetzung: Das DB-Register muß die Bankadresse der Tabelle enthalten ($40).

```
ASL              ; Funktionsnummer mal 2
TAX              ; X-Reg <- Fkt-Nummer * 2
JMP ($12ØØ,X)
```

Jeder Eintrag der Tabelle umfaßt zwei Bytes – daher ist eine Multiplikation mit demselben Faktor notwendig. Wenn der Akku den Wert 0 hat, bekommt X ebenfalls den Wert 0, es folgt ein Sprung zu PB/17AB. Hat der Akku den Wert 1, dann bekommt das X-Register den Wert 2, es wird zu PB/1802 gesprungen usw.
Wie bereits durch die Schreibweise PB/xxxx angedeutet, wird der Inhalt des PB-Registers dabei nicht verändert.
Dieselbe Adressierungsart ist auch für den Befehl JSR möglich. Der Rumpf eines menügesteuerten Programms könnte damit in etwa so aussehen:

```
     PHK        ; PB auf den Stack
     PLB        ; und in DB
     JSR ...    ; Initialisierung
     JSR ...    ; Menü-Ausgabe
XX1  JSR ...    ; Eingabe der Auswahl
     JSR ...    ; Eingabe -> Fkt-Nummer
     BCS XX1    ; -> ungültig
     ASL
     TAX
     JSR (TABELLE,X)
     BIT QUIT   ; QUIT-Funktion?
     BNE XX1    ; -> nein
XX2  <...>      ; Programmende
```

```
TABELLE EQU *
     DW FUNCØ   ; Adresse Funktion Ø
     DW FUNC1   ; Adresse Funktion 1
```

Die Voraussetzung, daß das DB-Register die Bankadresse der Tabelle enthält, wird hier durch einen einfachen Trick erfüllt (dessen Besprechung in diesem Abschnitt eigentlich nichts zu suchen hat): Zu Beginn des Programms wird das Programmbank-Register auf den Stack gebracht („PHK"), danach wird sein Wert wieder vom Stack geholt – und zwar in das Datenbank-Register („PLB"). Da beide Register immer dieselbe Breite haben (8 Bits), funktioniert dieses Verfahren immer. Um es genau zu nehmen, ist es die einzige Möglichkeit – dieser „Trick" findet sich zu Anfang sämtlicher bis dato von Apple, Inc. für den IIgs gelieferten Demonstrationsprogramme.

6.2.5. Erweiterte Stackoperationen – r,S und (r,S),Y

Die erste dieser beiden Adressierungsarten ist lediglich ein bequemerer Weg, an den Inhalt des Stacks ohne Veränderung des Stackpointers heranzukommen. „r,S" steht für „relativ zum Inhalt von SP" und läßt sich am besten durch ein Beispiel erklären:

```
LDX #$238B       ; (Indexregister 16 Bits)
PHX              ; erniedrigt SP um 2
LDA $Ø1,S        ; Akku <- Wert $238B
```

Auf den Befehlscode folgt ein Byte „Offset", das zum momentanen Stand des Stackpointers addiert wird. Das Ergebnis dieser Addition ist die Adresse der Speicherstelle(n) in Bank 00, mit deren Inhalt der Prozessor arbeitet.

Der Stackpointer wird für jede „Push"-Operation erniedrigt und zeigt immer auf die nächste freie Adresse innerhalb des Stacks. Ein Befehl wie „LDA $01,S" liest somit den zuletzt auf den Stack gebrachten Wert. Dasselbe ließe sich auch durch die folgenden Befehle erreichen:

```
TSX              ; X-Reg <- Stackpointer
LDA $ØØØØØ1,X
```

Achtung: Hier muß „absolut lang" adressiert werden – LDA $00,X benutzt das D-Register und funktioniert nur, wenn D den Inhalt $0000 hat; LDA $0000,X benutzt DB und hat nur dann das gewünschte Ergebnis, wenn DB auf $00 gesetzt ist.

Bevor wir uns mit der praktischen Anwendung und eventuellen Problemen dieser Adressierungsart beschäftigen, sehen wir uns die zweite Möglichkeit der Arbeit über den Stackpointer an:
Auf den Befehlscode folgt ein Byte „Offset", das zum momentanen Inhalt des Stackpointers addiert wird. Das Ergebnis dieser Addition wird als Adresse innerhalb der Bank 00 interpretiert. Ab dieser Adresse werden zwei Bytes gelesen. Zu dem gelesenen Wert wird der Inhalt des Y-Registers addiert. Das Ergebnis dieser (zweiten) Addition ist eine „absolute" Adresse innerhalb der Bank, auf die das DB-Register zeigt. Der Prozessor arbeitet mit den Werten, die er ab dieser Adresse aus dem Speicher liest.

Andersherum gesagt: Der Prozessor liest einen Pointer vom Stack und wendet die Adressierungsart „(Zp),Y" an – nur daß sich dieser Pointer eben nicht auf der Seite 0 befinden muß. Wir verwenden das zu „r,S" gezeigte Beispiel noch einmal:

```
LDX #$238B       ; 16 Bits!
PHX              ; erniedrigt SP um 2
LDA ($Ø1,S),Y
```

Wenn das Y-Register den Inhalt $0000 hat, dann lädt diese Befehlsfolge den *Inhalt* der Speicherstellen DB/238B und DB/238C in den Akku (also nicht den *Wert* $238B). Hat Y den Wert $0001, dann werden DB/238C und DB/238D gelesen, für Y = $0002 die Speicherstellen DB/238D und DB/238E usw.

Mögliche Probleme und Grenzen dieser beiden Adressierungsarten sind:

- Bei der Adressierung über den SP wird ein Überlauf in die Bank $01 hinein nicht berücksichtigt. Ein Befehl wie „LDA $F0,S" führt zur Adressierung der Speicherstellen $00/00xx, wenn SP einen Wert größer $FF0F hat. Dasselbe gilt für „LDA ($F0,S),Y": Wenn SP und „Offset" zusammen einen Wert größer $FFFF ergeben, wird der Pointer von der Adresse $00/00xx gelesen – und nicht von $01/00xx.
- Jede Routine, die „relativ" auf den Stack zugreift, muß „wissen", ob sie mit JSR oder JSL aufgerufen worden ist. Im ersten Fall liegt eine 2-Byte-Returnadresse vor, im zweiten Fall sind es drei Bytes (vgl. 3.1.).

Kommen wir endlich zur praktischen Anwendung: Diese beiden Adressierungsarten machen den 65816 ideal zur Implementation von Hochsprachen, die Parameter mit Hilfe des Stacks übergeben. Eine in Pascal formulierte Funktion und ihr Aufruf, also z.B.:

```
Function TX (Integer): Integer;

VAR_X:= TX(142);
```

läßt sich praktisch 1:1 in Assembler übersetzen. Zuerst der Aufruf:

```
LDA #$ØØ8E       ; Akku <- 142 dezimal
PHA              ; auf den Stack
JSR TX           ; Aufruf TX, liefert
PLA              ; Ergebnis auf dem Stack
STA VAR_X
```

Die Funktion TX benutzt die Adressierungsart „r,S" gleich zweifach, nämlich zum Lesen des übergebenen Wertes und zur Speicherung des Ergebnisses auf dem Stack:

```
LDA $Ø3,S        ; Akku <- Wert $ØØ8E
<...>            ; Rechenoperationen
<...>            ; mit diesem Parameter
STA $Ø3,S        ; Ergebnis auf den Stack
RTS
```

Innerhalb der Funktion TX wird über „$03,S" auf den Parameter zugegriffen – die Bytes „$01,S" und „$02,S" sind durch die Returnadresse belegt.

```
         A+4    $ØØ            <- $Ø4,S
         A+3    $8E            <- $Ø3,S
         A+2    RTS-Adresse    <- $Ø2,S
         A+1    RTS-Adresse    <- $Ø1,S
Adresse  A      ........       <- SP
```

Wie sieht ein Aufruf aus, bei dem die Anzahl der übergebenen Werte nicht mit der Anzahl der zurückgelieferten Werte übereinstimmt? Mit dem 65816 geht das ebenfalls erstaunlich einfach. Der Aufruf einer Prozedur wie

```
Procedure P1 (x,y: Integer);
```

besteht aus dem „Pushen" der benötigten Parameter:

```
LDA x           ; Akku <- Wert von x
PHA             ; auf den Stack
LDA y           ; Akku <- Wert von y
PHA             ; auf den Stack
JSR P1          ; Aufruf
PLA             ; y und x werden nicht
PLA             ; mehr gebraucht
<...>           ; weiter im Programm
```

Innerhalb von P1 werden die Parameter x und y vom Stack geholt:

```
LDA $Ø3,S       ; Achtung: das ist der
TAX             ; Wert von y!
LDA $Ø5,S       ; Parameter x
<...>           ; x und y bleiben
RTS             ; auf dem Stack
```

Eine Alternative, die das Entfernen der Parameter durch die Hauptroutine erspart, könnte in der Korrektur von SP durch die Routine P1 selbst bestehen, also

```
<...>           ; Rechenoperationen
PLX             ; Returnadresse!
TSC             ; Akku <- Stackpointer
CLC             ; um 6 erhöhen (x, y und
ADC #$ØØØ6      ; 2 Byte RTS-Adresse)
TCS             ; Stackpointer <- Akku
PHX             ; RTS-Adresse wieder
RTS             ; darauf und Ende
```

Dazu muß die Routine natürlich auf jeden Fall „wissen", wieviele Bytes des Stacks durch die übergebenen Parameter belegt waren.

Diese Art der Parameterübergabe wird auch als *Wert-Übergabe* bezeichnet – der Stack enthält eine Kopie des übergebenen Parameterwertes. Die andere Möglichkeit ist die Veränderung einer Variablen durch einen Funktions- oder Prozeduraufruf. In diesem Fall enthält der Stack nicht eine Kopie des Wertes, sondern die *Adresse* der Variablen selbst. Eine Beispielprozedur und ihr Aufruf:

```
Procedure SET_X (VAR x: Integer);

SET_X(Z);     {setzt Z neu }
```

Auch hier läßt sich der Ablauf mehr oder weniger 1:1 in Assembler umsetzen. Zuerst der Aufruf von SET_X:

```
LDA #z          ; Adresse(!) von z
PHA
JSR SET_X       ; Aufruf
PLA             ; nicht mehr gebraucht
<...>           ; weiter im Programm
```

Innerhalb der Prozedur SET_X kommt die zweite Stack-Adressierung des 65816 zum Einsatz: Da hier nicht der Wert, sondern die Adresse der Variablen z übergeben wurde, wird der Parameter als Pointer behandelt.

```
LDY #$ØØØØ
LDA ($Ø3,S),Y   ; Akku <- Wert von z
<...>           ; Operationen
STA ($Ø3,S),Y   ; z neu gesetzt
RTS
```

Solange es sich bei dem VAR-Parameter nur um einen 16-Bit-Wert handelt, mag der notwendige Befehl „LDY #$0000" etwas umständlich erscheinen. Anders sieht es dagegen aus, wenn z.B. mit einem Array gearbeitet wird, das aus einer Vielzahl von Elementen besteht. Mit einer Schleife wie

```
    LDY #$ØØØØ
XX1 LDA ($Ø3,S),Y     ; hat das Element
    CMP $Ø5,S         ; den gesuchten
    BEQ XX3           ; Wert (Parm #2)?
    INY
    TYA               ; Suchgrenze er-
    CMP $Ø7,S         ; reicht (Parm #3)?
    BCC XX1
XX2 <...>             ; -> nicht gefunden
XX3 <...>             ; -> gefunden
```

läßt sich ein beliebiges Datenfeld (Adresse: Parameter 1) nach einem bestimmten Wert (Parameter 2) absuchen. Parameter 3 gibt in diesem Beispiel die Obergrenze für die Suche vor.

6.2.6. PEA, PEI und PER

Um die Arbeit mit Stack-Parametern noch weiter zu erleichtern, kennt der 65816 drei weitere Befehle, die einen Wert ohne jede Beteiligung eines Registers auf den Stack bringen. Achtung: Es werden grundsätzlich zwei Bytes (d.h. ein 16-Bit-Wert) auf dem Stack abgelegt – unabhängig davon, auf welche Breite Akku und Indexregister gesetzt sind. Die Flags des P-Registers werden durch diese Operationen nicht verändert.

PEA („Push Effective Address") wird von einem 16-Bit-Operanden gefolgt, dessen Wert auf den Stack gebracht wird. Die bereits gezeigte Befehlsfolge

```
LDA #x          ; Adresse von x
PHA             ; auf den Stack
```

läßt sich damit durch einen einzigen Befehl ersetzen:

```
PEA #x          ; "pusht" Adresse von x
```

Ohne die Verwendung von Symbolen wird dieser Befehl vielleicht etwas deutlicher: Ein Befehl wie „PEA $17B4" bringt die Bytes $B4 $17 auf den Stack; $B4 steht danach auf der Adresse $01,S; $17 auf der Adresse $02,S.

Abgesehen von der Parameterübergabe läßt sich PEA auch noch für einen ganz anderen Zweck einsetzen – nämlich die indizierte Adressierung eines Datenfeldes ohne Benutzung der momentanen „Seite Null". Beispiel:

```
PEA #FELD       ; "pusht" Adresse v.FELD
LDY #ØØØØ
LDA ($Ø1,S),Y
PLA             ; nicht vergessen!
```

Über den Inhalt des Y-Registers lassen sich also die Elemente von FELD adressieren. Die gezeigte Befehlsfolge ist analog zu

Sammeldisk #28

Diskette mit allen DB-Hilfsprogrammen und einer Auswahl der Quelltexte sowie zwei Manuals zum Dateiprogramm DB-Meister.

Fortsetzungspreis DM 20,–.

Achtung: Da nur noch begrenzte Mengen der Original-Manuals vorrätig sind, erfolgt kein automatischer Versand an die Fortsetzungsbezieher, sondern Auslieferung in der Reihenfolge des Bestelleingangs, solange Vorrat reicht. Bitte bei Bestellung Fortsetzungsbezieher-Kundennummer angeben. Keine Lieferung an Nicht-Fortsetzungsbezieher!

T.DB-HELFER.OBJ
DB-HELFER.OBJ
T.DB-BILDSCHIRM
DB-BILDSCHIRM
T.DB-PRINTER
DB-PRINTER

DB-PFLEGER
DB-SORTER
DB-SORTER2
DB-FILTER
DB-DRUCKER

DB-HELFER
DB-MATCH-AENDERER
DB-MATCH-AENDERER.OBJ
DB-LESER
DB-LESER.OBJ
DB-LOESCHER
DB-STATUS
DB-TAUSEND
DB-DRUCKABBRUCH
DB-NUMMERNSORT
DB-BRIEFER.OBJ
DB-BRIEFER1.OBJ
DB-BRIEFER2.OBJ
DB-BRIEF

T.DB-MATCH-AENDERER.OBJ
T.DB-LESER.OBJ

DB-BAUER.ANLEITUNG
DB-BAUER.OBJ

RAMDISK-S4D1

DB-RECHNER-MUSTER

Hüthig Software Service
Postfach 10 28 69 · 6900 Heidelberg 1

```
LDA #FELD        ; Adresse von FELD
STA PTR          ; auf der "Seite Ø"
LDY #ØØØØ
LDA (PTR),Y
```

PEI („Push Effective Indirect address") geht einen Schritt weiter: Auf den Befehlscode folgt eine „Nullseitenadresse". Ab dieser Adresse werden zwei Bytes gelesen, der gelesene Wert wird auf den Stack gebracht. Beispiel: Wenn D den Wert $0100 hat und die Speicherstellen $00/0115 und $00/0116 die Werte $09 und $FB haben, dann legt der Befehl

```
PEI $15
```

die Werte $09 $FB auf dem Stack ab. $09 steht danach auf der Adresse $01,S; $FB steht auf der Adresse $02,S. Über diesen Befehl kann man also einen Pointer auf den Stack bringen.

PER („Push Effective PC Relative address") wird genauso wie PEA von einem 16-Bit-Operanden gefolgt. Der Prozessor liest diesen Wert und addiert ihn zu dem Wert, den der Programmzähler (PC) vor der Ausführung des nächsten Befehls hat (= momentaner Stand des PC plus 2). Darüber läßt sich eine Adresse auf den Stack bringen, die „soundsoweit" vom momentan laufenden Programmpunkt entfernt ist. Beispiel: Wenn sich auf der Adresse $xx/2000 der Befehl

```
PER $ØØØØ
```

befindet, dann werden die Bytes $02 $20 (also $2000 plus 2) auf den Stack gebracht. Sämtliche mir bekannten Assembler (einschließlich des Monitorprogramms des IIgs) behandeln das Argument eines PER-Befehls allerdings etwas anders: Die Eingabe des Befehls „PER $0000" kodiert die momentane Entfernung des PC vom angegebenen Wert. Wenn auf der Adresse $xx/2000 der Befehl „PER $2002" eingegeben wird, erzeugt der Assembler die Bytes $62 (PER-Befehlscode) $00 $00; die Eingabe von „PER $0000" erzeugt $62 $FD $DF – also die Entfernung des PC von der Adresse $0000 ($2002 plus $DFFD = $0000 mit nicht berücksichtigtem Überlauf).
Das folgende Beispiel hat exakt dasselbe Ergebnis wie das zum PEA-Befehl gezeigte – allerdings nur dann, wenn Programm- und Datenbankregister denselben Inhalt haben:

```
PER #FELD
LDY #ØØØØ
LDA ($Ø1,S),Y
```

Der Assembler kodiert hier den „Abstand" des Datenbereichs FELD zum Programmcode. Bei der Ausführung des Programms liest der Prozessor diesen Wert und addiert ihn zum momentanen Stand des PC hinzu. Die Folge: Die Adressierung von FELD ist unabhängig von der tatsächlichen Adreßlage des Programms, weil nur der (relative) „Abstand" zwischen Programmcode und Daten eine Rolle spielt.
Theoretisch ist es über PER möglich, ein Programm komplett von seiner Adreßlage unabhängig zu machen. Praktisch alle absoluten Sprünge (JMP) dürften sich durch lange relative Sprünge (BRL) ersetzen lassen. Ein relativer Unterprogrammaufruf ist nicht direkt implementiert – man kann ihn aber mit PER folgendermaßen ersetzen:

```
PER RET1-1   ; RTS-Adresse
BRL ...      ; "Aufruf" Unterprog.
RET1 ...     ; hier geht's weiter
```

Der Assembler kodiert für „PER RET1-1" die Bytes $62 (PER-Befehlscode) $03 $00 – folglich wird der Prozessor bei der Programmausführung den momentanen Stand des PC plus $0005 auf den Stack bringen. Das mit BRL „aufgerufene" Unterprogramm sollte mit einem RTS enden: Der Prozessor liest die Returnadresse vom Stack, erhöht den gelesenen Wert um eins und setzt ihn in den PC ein. Beispiel: Wenn das erste Byte des PER-Befehls auf der Adresse $2000 steht, wird der Wert $2005 auf den Stack gebracht; ein RTS setzt die Programmausführung ab der Adresse $2006 fort. Das Ergebnis ist exakt dasselbe wie ein normaler Unterprogrammaufruf mit JSR – nur daß das Programm auf einer beliebigen Adresse lauffähig ist.

6.2.7. Blocktransfers

Der 65816 verfügt über zwei Befehle, mit denen sich ganze Datenblöcke kopieren oder verschieben lassen. Blocktransfers sind sowohl innerhalb einer Datenbank als auch zwischen zwei verschiedenen Datenbanken möglich, Quell- und Zielgebiet können sich überlappen (müssen es aber nicht). Maximal können 64K pro Blocktransfer-Befehl kopiert/verschoben werden.
Wie bei ähnlichen Befehlen anderer Prozessoren (z.B. dem Z80-Befehl LDIR) erwartet auch der 65816 eine ganze Reihe von Vorbereitungen, bevor ein Blocktransfer ausgeführt werden kann:
– A, X und Y müssen auf 16 Bit Breite gesetzt sein;
– das Y-Register enthält die „absolute" Adresse innerhalb der Ziel-Bank;
– das X-Register enthält die „absolute" Adresse innerhalb der Quell-Bank;
– der Akku enthält die Anzahl der zu übertragenden Bytes (nicht: 16-Bit-Worte) minus 1;
– auf den Befehlscode für „MVP" oder „MVN" folgen zwei Bytes: das erste enthält die Banknummer der Zielbank, das zweite die der Quellbank.
Nach Ausführung des Befehls wird das DB-Register auf die Zielbank gesetzt, der Akku enthält den Wert $FFFF.

Der Befehl MVN („Move negative") erhöht X und Y während des Transfers, X und Y müssen also auf den *Beginn* von Ziel- und Quellbereich gesetzt sein. Nach Ausführung des Befehls enthalten sie die Endadresse von Ziel- bzw. Quellbereich plus 1. Der Name „Move negative" soll anzeigen, daß diese Transferart für Kopien in Richtung absteigender Speicheradressen vorgesehen ist.
Der Befehl MVP („Move positive") arbeitet in der umgekehrten Richtung: X und Y werden während des Transfers erniedrigt, müssen also vorher auf das *Ende* von Ziel- und Quellbereich gesetzt sein. Nach Ausführung des Befehls enthalten sie die Anfangsadresse von Ziel- bzw. Quellbereich minus 1. Vorgesehen ist diese Transferart für Kopien in Richtung aufsteigender Speicheradressen.

Solange sich Ziel- und Quellbereich eines Transfers nicht überlappen, spielt es keine Rolle, welcher der beiden Befehle verwendet wird. Das erste Beispiel kopiert den Bereich $02/4000..$02/54FF nach $04/2000..$04/34FF und benutzt dazu den Befehl MVN:

```
LDX #$4ØØØ      ; Quelladresse (Beginn)
LDY #$2ØØØ      ; Zieladresse (Beginn)
LDA #$14FF      ; $15ØØ Bytes
MVN $Ø2Ø4       ; von Bank $Ø2 nach $Ø4
```

Achtung: Das Argument von MVN wird als 16-Bit-Wort interpretiert – die tatsächliche Reihenfolge der Bytes im Speicher ist <Befehlscode> $04 $02.
Der Akku hat nach Ausführung dieser Befehle den Inhalt $FFFF, das X-Register den Wert $5500, Y steht auf $3500, DB hat den Wert $04. Dieselbe Operation läßt sich auch über MVP durchführen, weil sich Ziel- und Quellbereich nicht überlappen:

```
LDX #$54FF      ; Quelladresse (Ende)
LDY #$34FF      ; Zieladresse (Ende)
LDA #$14FF      ; $15ØØ Bytes
MVP $Ø2Ø4       ; von Bank $Ø2 nach $Ø4
```

Hier haben Akku und DB nach Ausführung des Befehls ebenfalls die Werte $FFFF bzw. $04; X und Y haben die Inhalte $3FFF (Quellenstart-1) bzw. $1FFF (Zielstart-1).

Zu guter Letzt noch eine bewußt *falsche* Verwendung von MVN, mit der sich eine Grafikseite des Apple sehr einfach auf einen bestimmten Wert löschen läßt:

```
LDA #$AAAA      ; dieses Wort wird
STA $ØØ2ØØØ     ; vervielfacht
LDX #$2ØØØ      ; Quelladresse (Start)
LDY #$2ØØ2      ; Zieladresse (Start)
LDA #$1FFD      ; $1FFE Bytes
MVN $ØØØØ       ; von Bank $ØØ nach $ØØ
```

Hier liest der Prozessor zuerst die Speicherstelle $00/2000 und speichert ihren Wert in $00/2002. X und Y werden um eins erhöht, der Prozeß wird für die Adressen $00/2001 und $00/2003 wiederholt. X und Y werden erneut erhöht, der Prozessor liest nun die Speicherstelle $2002 – also die Adresse, die im ersten Schritt des Kopiervorgangs mit dem Wert $AA beschrieben wurde. Dasselbe gilt für $00/2003 und den Transfer auf $00/2005 usw. Auf diese Weise pflanzt sich der Wert $AA durch die ganze Kopieraktion hindurch fort.

Von Blocktransfer-Befehlen wird eigentlich erwartet, daß sie wesentlich schneller arbeiten als eine aus „normalen" Befehlen zusammengesetzte Schleife. Für den 65816 trifft das nur dann zu, wenn der Transfer von einer Datenbank zu einer anderen stattfindet. Bei Tricks wie dem zuletzt beschriebenen fällt das dauernde interne Umladen des DB-Registers zeitlich so sehr ins Gewicht, daß eine extrem optimierte Schleife rund 20% schneller ist.

Literatur

[1] Jim Sather, Understanding the Apple IIe, erscheint in deutscher Übersetzung voraussichtlich Anfang 1987 bei Ampersand, Berlin
[2] Eyes/Lichty, Programming the 65816, Prentice Hall, 1986

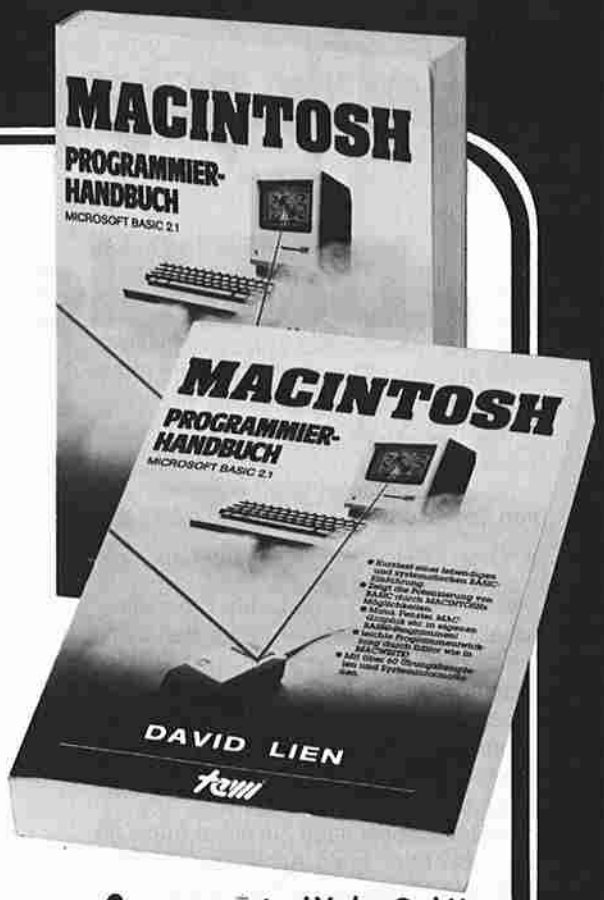
MS BASIC ist eine höchst moderne und leicht erlernbare Programmiersprache!

210 reservierte Begriffe...Programmsynthese aus Modulen (lokale Variablen, Wertübergaben mit COMMON)...Nachladen von Segmenten (CHAIN, mit Parameterübergabe)...Einbinden von MAC-Funktionen (Graphik, Maus, Fenster, Knöpfe usw.)...Fremddateizugriff (von MS-BASIC auf Dateien von z.B. MULTIPLAN, MACPAINT, WORD)...Programmablaufsteuerung (Ereigniserfassung wie ON TIMER, ON MOUSE)...Peripheriebefehle (wie COMI für DFÜ)...strukturierte BASIC-Programmierung ohne Zeilennummern...Gleitkommaarithmetik.....

MACINTOSH und MS BASIC bilden eine komfortable Programmierumgebung!

Mehrfachfenster für Simultanbeobachtung (z.B. Fenster 1: Hauptprogramm, Fenster 2: Unterprogramm; oder Fenster 1: Listing, Fenster 2: Ergebnisse)...Mausedition wie in Textprogrammen...Collagetechnik (Programmabschnitte ausschneiden und versetzen)...

HIER DER KURSTEXT einer lebendigen und systematischen BASIC-Einführung von dem US-Professor David Lien. Ein Text für das Selbststudium, das in anspruchsvolle BASIC-Programmierungen mit den Funktionen des MACINTOSH mündet. Ideal als Schultext.
450 Seiten, Softcover, DM 59,–

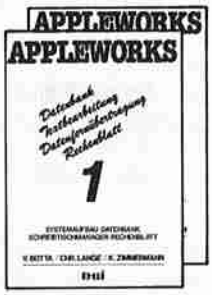


tewi Verlag GmbH
Theo-Prosel-Weg 1
8000 München 40

Weitere te-wi-Bücher

M68000 FAMILIE, 2 Bd.
Hilf/Nausch, ges. 968 Seiten
Einzige Motorola-authentische Darstellung von CPU-68000-Architektur, Programmierung, Systemaufbauten. Behandelt alle 68000-Bausteine sowie 68020, 68881.
Bd 1, Grundlagen + Architektur, 568 Seiten, DM 79,–
Bd 2, Anwendung und Bausteine, 400 Seiten, DM 69,–

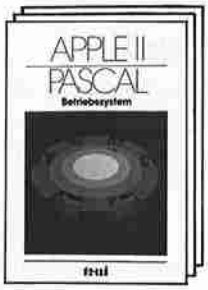
Das APPLE II/II+/IIe/IIc-Handbuch
L. Poole
Erst mit Hilfe dieses Leitfadens werden Sie Ihren Apple II erfolgreich einsetzen, denn Text und Bildmaterial gehen weit über das hinaus, was herstellerseitig an Literatur angeboten wird.
Neu überarbeitet und jetzt um die spezifischen Eigenheiten der Modelle **IIe** und **IIc** erweitert. 472 Seiten, Softcover, DM 66,–



APPLEWORKS integriert in APPLE II, IIe, IIc die Funktionen eines modernen Schreibtisches: Textverarbeitung, Datenbank, Rechenblatt, Datenfernübertragung. Sämtliche System/Anwendungsfragen in 2 Bänden.
Von Botta/Lange/Zimmermann, je 264 Seiten und je DM 49,–

DAS C-BUCH. **NEU**
Textbuch für C-Kurse und C-Anwendungen auf PCs. Beschreibt sämtliche Konstrukte der C-Sprache unter den Betriebssystemen MS DOS, CP/M, ISIS, UNIX und für die C-Compiler von MS, DR, LATTICE, INTEL. Didaktisch und typographisch außergewöhnlich. Mit über 100 lauffähigen Beispielprogrammen für PCs. Zeigt Realisierungen neuester Softwarestrategien in „C".
Von Herold/Unger. Herbst 86.
Etwa 500 Seiten. Softcover. DM 79,–

UMWELTDYNAMIK
30 Programme für kybernetische Umwelterfahrungen auf allen BASIC-Rechnern. Das Buch enthält beides: Ein Programmsystem zur Simulation eigener Problemformulierungen und 29 kommentierte Modellbeispiele wie Baumsterben, Heizungsbedarf, Nahrungsketten usw. Prospekt anfordern.
Von Hartmut Bossel, 480 Seiten, Softcover, DM 59,–



Erstes deutsches Referenzwerk
sämtlicher Befehle und Systemroutinen von Apple II, IIplus, IIe
APPLE II PASCAL
Betriebssystem, 272 S., DM 49,–
Sprache, 216 S., DM 39,–
Pascal 1.2 Addendum, 112 S., DM 36,–

Grundlagenbuch, Bestseller APPLE II PASCAL.
Eine praktische Anleitung, 544 S., DM 59,–

tm 4601

Double-Hires-Grafik-Paket

von Lothar Maier

1. Überblick

Dieses für DOS 3.3 gedachte Programmpaket, das mehr als die Hälfte der Sammeldisk #27 einnimmt und deshalb hier nicht gelistet werden kann, dient zur grafischen Aufbereitung von Anwenderprogrammen. Auf einfache Weise können übersichtliche Bildschirmdarstellungen erzeugt werden.
Zum Betrieb benötigen Sie einen Apple IIe mit 128K oder einen Apple IIc, ein Laufwerk und für die Utility-Programme einen Joystick.
Das Programmpaket unterstützt keine Farbgrafik. Sie kann jedoch mit entsprechender Positionierung der Grafiksymbole erreicht werden. Das Treiberprogramm kann mit Hilfe der Dokumentation einfach erweitert oder angepaßt werden.

Hier ein kurzer Überblick der zur Verfügung stehenden Befehle:
– Fenster können bei Bedarf erzeugt oder zwischengespeichert werden.
– Schrift kann vergrößert, invertiert oder in 90-Grad-Schritten gedreht ausgegeben werden.
– Menüs oder Funktionen können durch grafische Symbole (Icons) dargstellt und ausgelöst werden.
– Grafiksymbole können für Simulationen bewegt werden.
– Das Verschieben eines Fensterinhaltes ist punktweise in alle 4 Richtungen möglich.
– Sämtliche Befehle können von BASIC mit dem Ampersand-Befehl (&) ausgeführt werden.
– Treiberprogramm, Symbole und Zeichensätze befinden sich nahezu komplett im AUX-Memory.
– Zwei frei definierbare Zeichensätze stehen zur Verfügung, weitere können gepokt werden.
– Alle wichtigen CTRL-Funktionen stehen bei der Textausgabe zur Verfügung (Scrollen nach oben/unten, CLS, CLEOS, CLEOL...).
– Das Zeichnen von Punkten und Linien wird unterstützt.

2. Utility-Programme

Booten Sie DOS 3.3 oder Diversi-DOS, legen Sie die Sammeldisk #27 ein und starten Sie das Demo mit „EXEC DEMO-.START", das Ihnen eine Reihe von Befehlen demonstriert. Danach drücken Sie die RESET-Taste. Sie sollten sich jetzt im Textmodus befinden. Geben Sie nun übungshalber folgende Programmzeilen ein:

```
1Ø PRINT CHR$(12);:&D5,Ø,Ø,1
2Ø X=PDL(Ø):Y=PDL(1)
3Ø &bZ,Z,Z:IF Z THEN GOTO 5Ø
4Ø &M5,X,Y:GOTO 2Ø
5Ø &z:END
SAVE TEST
```

2.1. Blockeditor

Nachdem Sie Ihr Programm unter dem Namen „TEST" gespeichert haben, starten Sie die Utility-Programme mit „EXEC MENUE.DBLH.START". Sie wählen den Menüpunkt 1: BLOCKEDITOR. Nach einem kurzen Ladevorgang wird der Editor gestartet. Links oben sehen Sie ein zufälliges Muster, die rechte Hälfte des Bildschirms ist mit der Vergrößerung des Musters gefüllt. Drücken Sie die Tastenkombination ↑S. Das Muster wird gelöscht. Drücken Sie die Taste 3. Block 3 wird angewählt. Nehmen Sie den Joystick zur Hand. Drücken Sie Button 0, so wird der Punkt links oben in der vergrößerten Darstellung weiß. Drücken Sie Button 1, wird er wieder schwarz. Wenn Sie den Joystick bewegen, wandert ein X (Ihr Cursor) über den Bildschirm. Zeichnen Sie auf diese Weise ein kleines Bild. Drücken Sie die Taste S. Es öffnet sich ein kleines Fenster und Sie werden aufgefordert, einen Filenamen einzutippen. Geben Sie „TEST" ein. Wenn der Speichervorgang beendet ist, betätigen Sie die Tastenkombination ↑Q. So gelangen Sie wieder ins Menü.

2.2. Blocklinker

Wählen Sie jetzt Menüpunkt 2: BLOCKLINKER. Nach dem Ladevorgang drücken Sie die Taste L. Wieder öffnet sich ein kleines Fenster. Geben Sie „TEST" ein. Jetzt wird Ihr Bildchen geladen. Nach dem

Ladevorgang betätigen Sie die Taste 3. Sie müßten Ihr Bild links oben sehen. Drücken Sie die Taste D. Ihr Bild sollte sich jetzt auch im rechten Fenster befinden. Falls sich Ihr Bild *nicht* in der linken oberen Ecke des rechten Fensters befindet, bewegen Sie es bitte mit den Tasten I, J, K und M so in diese Ecke, daß es oben und links anstößt. Drücken Sie die Taste N. Sie werden nach einer Nummer gefragt, geben Sie bitte „5" ein. Bewegen Sie jetzt den Cursor auf die rechte untere Ecke Ihres Bildes. Drücken Sie jetzt die Taste C, anschließend die Tastenkombination ↑A. Der Inhalt des Fensters bewegt sich jetzt langsam nach rechts. Drücken Sie Taste S. Geben Sie jetzt wieder „TEST" ein. Ist der Speichervorgang beendet, drücken Sie ↑Q.

2.3. Startprogramm

Im Menü wählen Sie den Punkt 5: Startprogramm erstellen. Danach machen Sie die folgenden Eingaben:
Programmname:TEST
Soll das Programm über die Grafikseite geladen werden? (J/N) N
Name Zeichensatz 1:DEUTSCH
Name Zeichensatz 2:
Name der Shapetabelle:TEST
Name der Blocktabelle:
Soll der Grafiktreiber geladen und gestartet werden? (J/N) J

Wurden alle Angaben richtig gemacht, betätigen Sie die Taste S. Nachdem der Speichervorgang beendet ist, geben Sie E0 ein. Sie befinden sich nun auf dem gewohnten 80-Zeichenbildschirm.
Sehen Sie den Catalog Ihrer Diskette an. Folgende Files sollten sich unter anderem auf Ihrer Arbeitsdiskette befinden:

TEST.BLK (Binär, Ihr Bild)
TEST.START (Textfile, EXEC-File zum Laden der Programmteile)
TEST (Applesoft, Ihr erstes Programm)
TEST.TBL (Binär, Ihr Bild im Blockshape-Format)
DBLHLD (Binär, Laderoutine des Treibers)
DBLHTR (Binär, der Treiber selbst)

Starten Sie Ihr erstes Programm mit „EXEC TEST.START".
Sie können Ihr Bild mit dem Joystick auf dem Bildschirm bewegen. Wenn Sie Button 0 betätigen, wird das Programm beendet. Da jetzt alle benötigten Programme im Speicher stehen, kann das Programm mit „RUN" wieder gestartet werden.

2.4. Dateiübersicht

Natürlich sind die Möglichkeiten des Programmpaketes hiermit nicht ausgeschöpft, aber Sie haben jetzt einen kleinen Einblick in die Möglichkeiten bekommen, die Ihnen eröffnet werden. Auf der Diskette befindet sich ein Demonstrationsprogramm. Es kann, wie oben bereits gezeigt, mit „EXEC DEMO.START" ausgeführt werden. Es handelt sich um ein einfaches Applesoftprogramm, welches einige der neuen Grafikbefehle vorstellt. Das Programm BLOCKEDIT ist mit diversen REMs versehen, welche Ihnen die Programmierung unter DBLHTR näherbringen sollten.

Tabelle 1: Disketteninhalt

T.DBLHTR	Sourcecode Treiberroutine
DBLHTR	Hires-Treiber
T.DBLHLD	Sourcecode Laderoutine
DBLHLD	Laderoutine
T.CONVERT	Sourcecode HGR->Block
CONVERT	Übertragungsroutine
DEMO.START	Ladeprogramm für DEMO
DEMO	BASIC-Programm DEMO
DEMO.TBL	Blockshapes für DEMO
MENUE.DBLH.START	
MENUE.DBLH	Utility-Programme
BLOCKEDIT.DBLH	
BLOCKLINK.DBLH	
CHAREDIT.DBLH	
CONVERT.DBLH	
STARTER.DBLH	
DEUTSCH.SET	Zeichensätze
BYTE.SET	(BYTE.SET basiert auf Apple Toolkit)
COMP.BLK	Diverse Symbole
GRAF.BLK	

3. Funktionsübersicht über das Double-Hires-Ladeprogrammes (DBLHLD)

3.1. Allgemeines

DBLHLD dient zum Laden und Starten des Hirestreibers DBLHTR. MENUE.DBLH.START dient zum Starten der Utility-Programme. Es lädt den Treiber ab $4000 und verschiebt den Bereich $4000 bis $6FFF in die LC-Karte des AUX-Memory. Soll eine Programmerweiterung mitverschoben werden, muß diese vor dem Programmstart im entsprechenden Speicherbereich geladen sein. Zusätzlich wird ein „PR#3" durchgeführt und die Vektoren KSWL und CSWL auf den Treiber umgebogen. Die eigentliche Laderoutine endet mit „JMP $3D0", dem DOS-Kaltstart. Die Routine muß also über ein EXEC-File gestartet werden, um einen Programmabbruch nach dem Verschieben zu vermeiden. Dieser EXEC-File kann mit STARTER.DBLH erstellt werden.
Während der Programmausführung von DBLHTR werden die gesamten Softswitch-Aufgaben von DBLHLD gesteuert. Ferner wird mit dem Unterprogramm TRANSFER die Speichermanipulation AUX nach MAIN und umgekehrt ermöglicht. TRANSFER wird beim Laden und Speichern von Bildern, Blockshapes und Zeichensätzen benötigt. Bitte beachten Sie, daß beim ersten Programmstart *kein* Zeichensatz vorhanden ist.

3.2. TRANSFER

TRANSFER wird durch den Befehl „CALL 891,Anfang,Ende,Ziel,Richtung" aktiviert. Anfang, Ende und Ziel müssen im Bereich $800 bis $BFFF liegen. Die Variable Richtung darf die Werte 0 oder 1 annehmen, wobei 0 von AUX nach MAIN überträgt.

3.3. Wichtige Befehlsfolgen für TRANSFER

Speichern eines DBLH-Bildes:
CALL891,8192,16383,16384,0
BSAVE BILD,A$2000,L$4000

Laden eines DBLH-Bildes:
BLOAD BILD,A$2000
CALL891,16384,24575,8192,1

Speichern einer Blocktabelle:
&GA:IF A<0 THEN A=A+65535
CALL891,16384,A,16384,0
A=A-16384
BSAVE BLOCK.TBL,A$4000,LA

Laden einer Blocktabelle:
BLOAD BLOCK.TBL,A$4000
L=PEEK(43616)+PEEK(43617)*256
CALL891,16384,16384+L,16384,1
&H16384+L,0

Laden von 2 Zeichensätzen:
BLOAD SATZ1.SET,A$4000
BLOAD SATZ2.SET,A$4300
CALL891,16384,17910,2048,0

Bei den Befehlsfolgen wurde davon ausgegangen, daß der Hauptspeicher oberhalb $4000 frei ist. Es könnte jedoch auch jeder andere freie Speicherbereich verwendet werden.

4. Funktionsübersicht des Double-Hires-Treibers (DBLHTR)

4.1. Allgemeines

Das Programm DBLHTR dient zur Manipulation der doppelt hochauflösenden Grafik des Apple IIe oder IIc. Die Textaus- und -eingabe wird auf die Grafik umgelenkt, jedoch stehen die gewohnten Editorkommandos nicht mehr zur Verfügung. Die Programmerstellung sollte deshalb im normalen Textmodus vorgenommen werden. Die Ausgabe von CTRL-Zeichen löst verschiedene Funktionen aus, die in etwa den neuen IIe-ROMs entsprechen. Zur Erstellung von Grafiken steht ein leistungsfähiger Befehlssatz zur Verfügung. Dieser enthält „Hplot"-Befehle und erlaubt die einfache Manipulation von „Blockshapes". Zusätzlich wurde eine einfache Fenstertechnik implementiert, die das Speichern und Wiederherstellen von bis zu 15 Fenstern erlaubt.

Verwendete Speicherbereiche

Die Treiberroutinen befinden sich in der LC-Card des AUX-Mem und haben eine Länge von knapp 6K. Die beiden Zeichensätze der Textausgabe sind ab $800 im AUX-Mem und haben eine Länge von je $300 Bytes. Die Blockshapes und eventuell zwischengespeicherte Fenster sind ab $4000 ebenfalls im AUX-Memory. Die höchste für diese Anwendungen freie Speicherzelle ist $BFFF. Die Softswitch-Routinen befinden sich in der Seite 3 des MAIN-Memory. Durch diese Speichereinteilung steht nahezu der gesamte, bei Hiresanwendungen freie Speicherplatz zur Verfügung.

Zugriff

Der Zugriff auf die Double-Hires-Funktionen erfolgt über „&"-Befehle.

4.2. Funktion der CTRL-Codes bei Textausgabe

HEX	DEZ	ASCII	FUNKTION
Ø1	Ø1	CTRL-A	Fensterinhalt eine Zeile nach unten verschieben, Cursor in oberste Zeile
Ø2	Ø2	CTRL-B	Fensterinhalt eine Zeile nach oben verschieben, Cursor in unterste Zeile
Ø3	Ø3	CTRL-C	Der linke Fensterrand wird auf die aktuelle Cursorposition gesetzt
Ø5	Ø5	CTRL-E	Der linke Rand wird wiederhergestellt
Ø8	Ø8	CTRL-H	Cursor 1 Zeichen n. links
ØA	1Ø	CTRL-J	Cursor 1 Zeile nach unten
ØB	11	CTRL-K	Fenster ab Cursor löschen
ØC	12	CTRL-L	löscht F., Cursor Home
ØE	14	CTRL-N	Zeichenausgabe normal
ØF	15	CTRL-O	Zeichenausgabe invers
15	21	CTRL-U	Cursor 1 Zeichen n. rechts
16	22	CTRL-V	Fenster 1 Zeichen n. unten
17	23	CTRL-W	Fenster 1 Zeichen n. oben
18	24	CTRL-X	Zeichensatz 1
19	25	CTRL-Y	Cursor Home
1A	26	CTRL-Z	Zeile löschen
1B	27	CTRL-Ä	Zeichensatz 2
1D	28	CTRL-Ü	Zeile ab Cursor löschen
1E	29	CTRL-↑	Cursor 1 Zeile nach oben

4.3. Befehlsübersicht Grafik

Parameter	Bedeutung
X	Vertikale Position Ø...191
Y	Horizontale Position Ø...559
H	Horizontale Byteposition Ø...79
Ti	Tiefe Ø...191
Br	Breite Ø...79
A	Adresse Ø...65535
V	Übergabevariable Ø...65535
M	Zeichenmode Ø...3(Ø...2 bei Plot)

M=	Block	Plot	Vergrößerung
Ø	OR	OR	OR
1	EOR	EOR	EOR
2	SCHWARZ	SCHWARZ	SCHWARZ
3	STORE	–	–

R	Richtung Ø Ø Grad 1 9Ø Grad 2 18Ø Grad 3 27Ø Grad
W	Window Ø..15
B	Block Ø..63
O	Oben Ø..191
U	Unten 1..192(unterste Pos.+1)
L	Links Ø..79
R	Rechts 1..8Ø(letzte Spalte+1)
Z	Dummy, wird nicht verwendet
X*	Horizont. Vergrößerung Ø..255
Y*	Vertik. Vergrößerung Ø..255

4.3.1. Blockshapebefehle

Befehl	Parameter	Funktion
&D	B,X,Y,M	Block zeichnen
&M	B,X,Y	Block in alter Position löschen, an neuer Pos. zeichnen
&S	B,X,Y,Br,Ti	Block von Bildschirm in Tab. übertragen
&C	B,B	Zeiger von Block# nach Block# kopieren
&o	B,B	Block# nach Block# kopieren (oben und unten vertauscht)
&l	B,B	Block# nach Block# kopieren (links und rechts vertauscht)
&E	B,B,V	Überschneiden sich Block# und Block#, ist V>Ø
&G	V	V gibt aktuelles Speicherende aus
&H	V,Z	Speicherende wird auf V gesetzt

4.3.2. Punktbefehle

&P	X,Y,M	Punkt
&A	X,Y,M	Linie von letztem Punkt nach X,Y
&F	X,Y	Wenn Punkt weiß, wird V>Ø
&Y	X,Y	Hires-Cursor setzen

4.3.3. Fensterbefehle

&O	W	Fensterinhalt einen Punkt nach oben
&U	W	Fensterinhalt einen Punkt nach unten
&L	W	Fensterinhalt einen Punkt nach links
&R	W	Fensterinhalt einen Punkt nach rechts
&I		Fenster invertieren
&W	W	Fenster wählen
&Z	W,L,R,O,U	Fenster definieren
&w	W	Fensterhintergrund speichern
&r	W	Fensterhintergrund wiederherstellen

4.3.4. Speicherblockbefehle

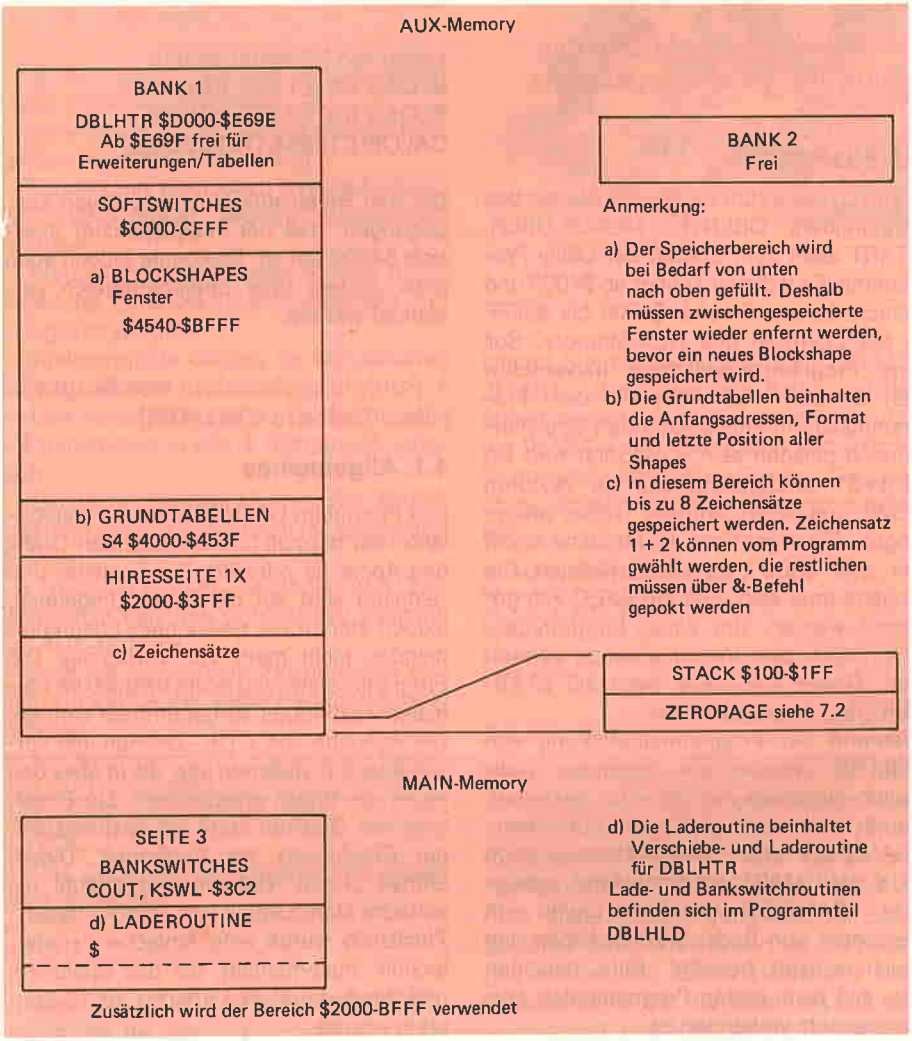
&J	A,H,Y,Br,Ti	Speicherinhalt nach Grafik
&K	A,H,Y,Br,Ti	Grafikausschnitt nach Speicher

4.3.5. Vergrößerungsbefehle

&c	C,X,Y,X*,Y*,M,R	Zeichen nach Grafik
&d	B,X,Y,X*,Y*,M,R	Block nach Grafik
&m	A,Br,Ti,X,Y, X*,Y*,M,R	Speicherinhalt nach Grafik

4.3.6. Sonstige Befehle

&§	H,Y	HTAB H,VTAB Y
&p	V,V,V,V	Paddleabfrage P3,P2,P1,PØ Bereich: V= -1...+1



&b	V,V,V	Buttonabfrage B2,B1,BØ Button gedrückt = 255
&s	A,V	Auxpoke
&g	A,Z,V	Auxpeek
&e		Sprung in Prg.erweiterung
&z		Bildschirm löschen (grau)

5. Erläuterungen

5.1. Control-Codes (s. 4.2.)

Die Texteingabe unterdrückt ESC und CTRL-U, da nicht vom Bildschirm kopiert werden kann. Die Geschwindigkeit der Textausgabe entspricht etwa der normalen 80-Zeichenausgabe, sofern nicht gescrollt werden muß. Beim Scrollen müssen bis zu 16K bewegt werden, was natürlich seine Zeit braucht.
HTAB, TAB(), VTAB und Kommatabulierung haben keine Funktion und werden durch den Befehl &§H,Y ersetzt. Die Umschaltung zwischen Mauszeichensatz und normalem Zeichensatz schaltet bei DBLHTR zwischen Zeichensatz 1 und 2 um.
CTRL-A und CTRL-B ermöglichen eine einfache Ausgabe von Texten, die länger als die Fensterhöhe sind. Beim Anfügen einer Zeile wird der Text nach oben oder nach unten verschoben. Die angefügten Zeilen dürfen den rechten Fensterrand nicht erreichen und müssen mit Semikolon enden.
CTRL-C und CTRL-E erlauben das Einrükken eines Absatzes. Zweimaliges CTRL-C ohne CTRL-E führt zum Verlust des linken Fensterrandes!

5.2. Blockshapes (s. 4.3.1.)

Blockshapes – in der Literatur auch als Rastershapes bezeichnete Grafiksymbole – dienen als schneller Ersatz für die gewohnten Apple-Shapes. Ein Blockshape ist ein Speicherbereich, der mit definierter Breite und Höhe in die Grafik übertragen wird. Um ein Blockshape beliebig auf dem Bildschirm plazieren zu können, muß es siebenmal im Speicher vorhanden sein (jeweils um einen Punkt nach rechts verschoben). Dies führt zu einem hohen Speicherbedarf.
Beispiel: Ein Shape mit der Breite von 5 Bytes und einer Höhe von 20 Punkten benötigt 840 Bytes! Allerdings befinden sich die kompletten Tabellen im AUX-Memory. Soll ein Blockshape nur in den Spaltenpositionen (0,7,14,21...) gezeichnet werden, so können die verschobenen Tabellen weggelassen werden. Speicherbedarf bei obigem Beispiel: 100 Bytes.

Das Übertragen eines Blockshapes auf den Grafikbildschirm erfordert die Nummer des Shapes (0...63), die Position und die Methode, mit der es gezeichnet werden soll. Erhalten Sie einen Bildschirm füllenden Unsinn, wollten Sie ein nicht definiertes Blockshape zeichnen. Erhalten Sie nichts, befinden Sie sich beim Zeichnen außerhalb des geöffneten Fensters, oder Sie haben den falschen Zeichenmode gewählt, z.B M=0 auf inversem Bildschirm. Eine weitere Fehlermöglichkeit ist das Verwenden einer unvollständigen Tabelle, z.B. wenn Sie das Shape nur in der Grundposition gespeichert haben und es nicht in einer Spaltenposition zeichnen. Im ungünstigsten Fall wird das Shape dann aus dem Softswitch-Bereich geholt. Verfügen Sie über einen Farbmonitor, werden Sie feststellen, daß ein Shape im Abstand von 4 Punkten die gleiche Farbe hat. Dies ist auf die etwas (sehr) eingeschränkten Farbmöglichkeiten des Apple zurückzuführen (Text wird beinahe unlesbar, also Farbe aus).

Sie haben vier Möglichkeiten, ein Blockshape auf den Bildschirm zu bringen:
1. OR entspricht dem DRAW-Befehl.
2. EOR entspricht dem XDRAW-Befehl.
3. BLACK zeichnet die gesetzten Punkte des Blockshapes schwarz (funktioniert nur auf weißem Grund).
4. STORE überträgt das Blockshape auf den Bildschirm und löscht den Hintergrund.

Es gibt verschiedene Möglichkeiten, ein Blockshape zu bewegen. Einfachste Methode: Sie verwenden den Befehl &MX,Y (genaue Funktion s.u.). Für eine möglichst flimmerfreie Bewegung sollten Sie den STORE-Mode verwenden, dieser erfordert jedoch einen schwarzen Rand um das Blockshape und läßt so nur die Bewegung um die Breite das Randes zu. Ein eventuell vorhandener Hintergrund wird mitgelöscht.

Die Bewegung eines Blockshapes wird auf folgende Weise ermöglicht: Löschen durch EOR an alter Position, Zeichnen mit EOR an neuer Position. Vor dem ersten &MX,Y muß das Blockshape irgendwo gezeichnet werden, um eine 1.Position zum Löschen zu haben. Zwischen den &MX,Y-Befehlen sollten die Fenster nicht gewechselt werden, da sonst unter Umständen das Blockshape nicht mehr gelöscht werden kann (falls es sich außerhalb des neuen Fensters befindet). Das Blockshape kann außerhalb eines Fensters bewegt werden, es bleibt jedoch unsichtbar.

Das Speichern eines Blockshapes in der Tabelle erfordert etwas Planung. Folgende Punkte sollten beachtet werden:
– Prüfung des noch vorhandenen Speicherplatzes (mit &G V)
– Horizontale Position (X): Das Blockshape wird entsprechend seiner Position in der Tabelle gespeichert. Das bedeutet, daß das Blockshape bei X=1 in der Tabelle mit Offset=1 gespeichert wird, was zur Folge hat, daß das Blockshape nur in den Positionen 1,8,15... gezeichnet werden kann. Der jeweilige Offset kann mit folgender Formel berechnet werden: Offset = X / 7 - INT (X / 7). Auf diese Weise ist es möglich, ein Shape zu erstellen, das sich in jeder Position ändert. (Ein „Alien" flattert bei horizontaler Bewegung.)
– Breite und Höhe müssen bei jedem Offset gleich sein. Der Rand (1 Byte) auf der rechten Seite muß mitgerechnet werden, wenn ein bewegbares Shape erstellt werden soll. Ein linker, rechter, oberer und unterer Rand muß je nach gewünschter Bewegung (mit DRAWMODE=3) hinzugefügt werden.
– Ein gespeichertes Blockshape kann nicht mehr gelöscht werden.
– Verwenden Sie die Programme BLOCKEDIT und BLOCKLINK zur Erstellung der Tabellen.

Mit dem Befehl &C1,2 wird die Tabelle von Blockshape #1 nach Blockshape #2 kopiert. So wird ermöglicht, z.B. mehrere gleichartige Symbole in verschiedene Richtungen zu bewegen. Bei diesem Befehl wird kein weiterer Speicherplatz benötigt.
Bei dem Befehl &o3,6 wird Blockshape #3 nach Blockshape #6 kopfstehend kopiert. Es wird jedoch nur der Offset #0 übertragen. Es wird eine neue Tabelle angelegt und somit auch der entsprechende Speicherplatz verbraucht.
Die Funktion &I ist wie &o, jedoch wird das Blockshape seitenverkehrt übertragen.
Mit &E wird eine Überschneidung zweier Blockshapes berechnet. Wenn eine Berührung stattfindet, wird V>0. Dies gilt natürlich auch bei nicht gesetzten Punkten oder bei durch den HOME-Befehl gelöschten Shapes.
Mit dem Befehl &G erhalten Sie das Speicherende der Blockshapetabellen. Wird V>32768, wird V negativ (s. FRE(0)). Die Tabellen dürfen bis $BFFF gespeichert werden, dahinter beginnt der Softswitch-Bereich. Dieser macht sich sehr unangenehm bemerkbar.
Mit &H kann das Speicherende neu definiert werden. Es ist sogar möglich, Blockshapes hinter den Grafiktreiber in die LC-Karte des AUX-Mem zu speichern. Sie müssen jedoch darauf achten, daß keine anderen Routinen überschrieben werden.

5.3. Punktbefehle (s. 4.3.2.)

Die Punktbefehle arbeiten etwa halb so schnell wie auf der HGR-Seite. Allerdings wird bei diesen Befehlen keine so hohe Geschwindigkeit benötigt.

Das Setzen von Einzelpunkten kann mit 3 Zeichenmethoden erfolgen:

1. OR entspricht dem Plotbefehl.
2. EOR: der Bildschirmpunkt wird invertiert.
3. BLACK: der Punkt wird gelöscht.

Der Befehl &A enspricht HPLOT TO. Er verbindet den letzten gesetzten Punkt mit den neue Koordinaten.
Sie können mit &F ermitteln, ob ein Hires-Punkt gesetzt oder gelöscht ist.
Das Setzen des Hires-Cursors wird benötigt, um eine beliebige Linie mit M=1 zeichnen zu können, da bei der Befehlsfolge „&P0,0,1:&A100,100,1" der erste Punkt wieder gelöscht wird (zweimaliges EOR).

5.4. Fensterbefehle (s. 4.3.3.)

Die Fenstertechnik erlaubt eine einfache Text- und Grafikausgabe. Beim Laden von DBLHTR werden alle 15 Fenster auf den vollen Bildschirm gesetzt.
Die Schiebebefehle erlauben die einfache Manipulation des Fensterinhalts. Bei diesen Befehlen muß das jeweilige Fenster nicht geöffnet sein. Alles, was den Fensterrand überschreitet, wird schwarz gelöscht.
Das Öffnen eines Fensters bewirkt die Neueinstellung der Bildschirmränder. Der Cursor wird auf die linke obere Ecke gesetzt. Das Löschen des Fensters muß mit CTRL-L erfolgen. Der Fensterhintergrund wird zerstört.
Beim Setzen eines Fensters ist darauf zu achten, daß die Höhe mindestens 8 Bit, die Breite mindestens 1 Zeichen betragen muß. Sollen die Befehle CTRL-A und CTRL-B eingesetzt werden, muß die Fensterhöhe durch 8 ohne Rest teilbar sein.
&w speichert den Hintergrund des Fensters und öffnet es anschließend. Es können mehrere Fenster hintereinander geöffnet werden. Sie müssen in der umgekehrten Reihenfolge wieder geschlossen werden. Solange ein Fenster gespeichert ist, sollten keine Blockshapes erstellt werden, da die Fenster ebenfalls nach dem Programmende gespeichert werden. Alle mit &wW geöffneten Fenster müssen mit &rW geschlossen werden. Das Schließen des Fensters stellt den Hintergrund wieder her, der beim Öffnen vorhanden war. Wird ein Fenster geschlossen, das nicht geöffnet war, wird mit Sicherheit etwas Seltsames passieren.

5.5. Speicherblockbefehle (s. 4.3.4.)

Diese Befehle erlauben das Speichern und Laden von Grafikausschnitten im Hauptspeicher. Es muß natürlich dafür Sorge getragen werden, daß kein Anwenderprogramm überschrieben wird. Es können horizontal nur ganze Spaltenpositionen (0...79) verwendet werden. Trotzdem sind diese Befehle zum Bildschirmaufbau sehr praktisch und in der Anwendung einfach.
Die Adresse darf sich nicht in der LC-Card befinden.
Das Zeichnen erfolgt immer mit XDRAW. Die Ausgabe ist recht schnell, auch beim Zeichnen eines kompletten Bildschirmes.

5.6. Vergrößerungsbefehle (s. 4.3.5.)

Die Vergrößerungsbefehle arbeiten relativ langsam. Wird die Vergrößerung entsprechend klein gewählt, kann jedoch ein etwas schnellerer Bildaufbau erreicht werden. Die Vergrößerung kann in 90-Grad-Schritten gedreht werden. Als besonders wichtige Anwendung möchte ich die Beschriftung von Achsen erwähnen.

5.7. Sonstige Befehle (s. 4.3.6.)

Die Paddleabfrage mit &p geschieht zwar analog, die Auswertung erfolgt jedoch digital. Es werden alle 4 Paddleeingänge abgefragt.
Mit &b werden alle 3 Buttoneingänge abgefragt und das mühsame Peeken erspart. Auxpeek und Auxpoke erlauben die Speichermanipulation im AUX-Mem bzw. in der Aux-LC.
Mit dem Befehl &e kann DBLHTR erweitert werden. Mit dem Sprung gelangt man an das Programmende von DBLHTR, wo sich die Erweiterung befinden sollte.
Mit &z wird der komplette Bildschirm „grau" gelöscht. Dies geschieht ohne Rücksicht auf Fenster.

6. Funktionsübersicht über die Utility-Programme

6.1. Allgemeines

Die Utility-Programme bestehen aus einem Menü und fünf Programmteilen. Sie dienen zum einfachen Erstellen und Ändern von Grafiksymbolen und Zeichensätzen.

Name	Funktion
BLOCKEDITOR	Grafiksymbolerstellung
BLOCKLINKER	Grafiksymbole zu Blockshapetabellen zs.setzen
ZEICHENSATZEDITOR	Zeichensatzerstellung
KONVERTER	HGR- od. DHGR-Bilder -> zu Symbolen od. Zeichen, die mit entsprechenden Editoren weiterverarbeitet werden können
STARTPROGRAMM	Erstellt aus DBLHLD, DBLHTR, Blockshapetab., Zeichensätzen und einem Anwenderprg. ein ablauffähiges Gesamtprg.

6.2. BLOCKEDITOR

Der Blockeditor erlaubt das Erstellen von bis zu zehn quadratischen Grafiksymbolen mit maximal 84 Punkten Breite und 40 Punkten Höhe. Die erstellten Symbole werden im MAIN-Memory gespeichert und können je nach Bedarf entweder direkt verwendet werden oder mit dem BLOCKLINKER zu Blockshapetables zusammengefaßt werden. Die vom BLOCKEDITOR erstellten Symbolgruppen können, im Gegensatz zu den Blockshapetables, geändert werden. Soll also eine Bilderbibliothek angelegt werden, muß dies mit dem BLOCKEDITOR gemacht werden. Die Symbolgruppen werden mit dem Suffix „.BLK" abgespeichert und benötigen auf der Diskette 20 Sektoren. Wollen Sie geometrische Figuren wie Kreis oder Oval erstellen, sollten Sie ein kurzes BASIC-Programm schreiben, welches die Figur auf HGR oder DHGR zeichnet, den Bildschirm abspeichern und mittels KONVERT Ihre Figur in den BLOCKEDITOR übertragen.

6.2.1. Display

In der linken oberen Ecke wird das bearbeitete Symbol in Orginalgröße angezeigt. Rechts daneben befindet sich die Symbolnummer und die Cursorposition. Darunter befindet sich die Befehlstabelle. Die rechte Bildschirmseite beinhaltet die vergrößerte Darstellung des Symbols sowie den X-Cursor.

6.2.2. Befehlsübersicht

Die Cursorsteuerung wird vom Joystick übernommen. Button 0 erzeugt einen weißen Punkt, Button 1 löscht einen weißen Punkt.

Die Tasten haben folgende Funktionen:

S öffnet das Speicherfenster. Es muß ein Filename eingegeben werden. Unter diesem Namen wird die Symbolgruppe auf dem Schreiblaufwerk gespeichert. Bei einem Fehler wird der Speichervorgang abgebrochen, das Programm jedoch fortgesetzt.
L öffnet das Ladefenster. Es muß ein Filename eingegeben werden. Dieser File wird vom Leselaufwerk in den Speicher geladen. Wird kein Filename oder ein nicht vorhandener Filename eingegeben, wird der Ladevorgang unterbrochen. Beim Laden wird der Blockspeicherinhalt gelöscht.
0...9: Mit den Zifferntasten kann ein Symbol angewählt werden. Das Symbol wird erst nach Anwählen eines anderen Symbols oder bei „S" in den Speicher übernommen. Wird das Programm beim Erstel-

len eines Symbols mit ↑C abgebrochen, ist das bearbeitete Symbol verloren.
↑**B**: Das angezeigte Symbol wird gelöscht.
↑**S**: Alle Symbole werden gelöscht.
↑**D**: Mit diesem Befehl wird das DOS-Fenster geöffnet. Sie können das Lese- und Schreib-Laufwerk wählen oder den Catalog des Leselaufwerkes ausgeben lassen.
↑**Q**: Das Menüprogramm wird aufgerufen. Beim Verlassen des Programmes wird der Symbolspeicher nicht gelöscht, es sei denn, Sie laden anschließend ein Bild oder einen Zeichensatz. So wird ermöglicht, daß Sie z. B. zwischen BLOCKEDITOR und KONVERTER umschalten können.

6.3. BLOCKLINKER

Der Blocklinker erlaubt das Erstellen von Blockshapetabellen (Blockshapetables) aus den vom BLOCKEDITOR kreierten Symbolen. Ein Blockshapetable kann bis zu 64 Blockshapes beliebiger Größe enthalten. Ein nachträgliches Ändern der erstellten Tabellen ist nicht möglich. Gespeichert werden die Tabellen mit dem Suffix „.TBL". Beim Erstellen der Tabellen ist darauf zu achten, daß der angezeigte freie Speicherbereich nicht überschritten wird, da sonst beim Speichern DOS überschrieben wird. Benötigen Sie eine längere Tabelle, müssen Sie den Blocklinker mit ↑C unterbrechen, die Länge der Tabelle feststellen, diese manuell ins MAIN-RAM übertragen und ebenso speichern. Weiterhin ist zu beachten, daß beim mehrmaligen Speichern des gleichen Shapes mit gleichem Offset das Shape nicht überschrieben, sondern einfach neu ans Tabellenende angehängt wird. Die Größe des Shapes muß bei verschiedenen Offsets gleich sein und sollte deshalb mit Bedacht gewählt werden.

6.3.1. Display

In der linken oberen Ecke befindet sich das Symbol, mit dem gerade gearbeitet wird. Diese Anzeige wird nur durch die Symbolwahl aktualisiert (Achtung beim Laden einer neuen Symbolgruppe!). Daneben befindet sich die Anzeige der Cursordaten und die Tastaturfunktionen für das Übertragen ins Shapefenster. Darunter befinden sich die restlichen Tastaturfunktionen. Das Shapefenster befindet sich in der rechten Hälfte des Bildschirmes. Über dem Shapefenster befinden sich die Anzeige des freien Speicherplatzes und die gerade eingestellte Größe des Shapes. Unter dem Fenster wird die Shapenummer und der Offset angezeigt. Der Cursor ist ein blinkender Punkt, welcher sich auch außerhalb des Shapefensters befinden kann (siehe Anzeige X= Y=; wird einer dieser Werte negativ, ist der Cursor weg).

6.3.2. Tastenbelegung

Die Cursorsteuerung erfolgt über den Joystick, die Schrittweite wird durch Gedrückthalten des Button 0 versiebenfacht. Mit Button 1 kann der Punkt unter dem Cursor invertiert werden.

Die Tasten haben folgende Funktionen:
S: Speichern der Blockshapetabelle
L: Laden einer Symbolgruppe
I,J,K,M: Der Inhalt des Shapefensters wird um einen Punkt in der jeweiligen Richtung verschoben. Bei Links- und Rechtsbewegungen wird der Offset geändert. Der Inhalt des Fensters, der den Rand überschreitet, wird schwarz gelöscht. Bevor Sie ein Shape speichern, sollten Sie es in die linke, obere Ecke verschieben, um die Größe möglichst klein zu halten.
↑**M (RET)**: Das Shape wird entsprechend der Shapenummer, dem Offset und mit der angezeigten Größe in die Tabelle übernommen. Rechts neben dem Fenster wird ein Punkt als Gedächtnisstütze angezeigt. Mit diesem Befehl sollten nur solche Shapes erstellt werden, die sich bei verschiedenen Offsets verändern. Eine weitere Möglichkeit besteht darin, bis zu 448 verschiedene Shapes in der Tabelle zu speichern, die mit der Shapenummer *und* dem Offset aufgerufen werden müssen.
↑**A**: Automatisches Speichern und Verschieben eines Shapes. Dies erzeugt ein komplettes Shape, das in jeder Bildschirmposition gezeichnet werden kann. Bitte beachten Sie hierbei besonders den freien Speicherplatz, die Position und die Größe des Shapes. Ist z.B. die Breite zu gering gewählt, verschwindet der hintere Teil des Shapes in den entsprechenden Positionen. Dies ist beim Erstellen der Tabelle nicht ersichtlich. Verwenden Sie deshalb die Funktionstaste „C".
N: Wahl der Shapenummer
O: Manuelle Offseteinstellung
B: Manuelle Breiteneinstellung
H: Manuelle Höheneinstellung
C: Die Cursordaten werden in Höhe und Breite umgerechnet und eingestellt. Bewegen Sie den Cursor hierzu auf den am weitesten rechts liegenden Punkt und danach in die unterste Zeile Ihres Shapes, und drücken Sie dann die Taste „C".
↑**L**: Löscht das komplette Shapefenster.

Übertragen eines Symbols ins Shapefenster:
D: Das Symbol wird mit dem Fensterinhalt ODER verknüpft.
X: Das Symbol wird EXKLUSIV ODER verknüpft.
F: Die Stellen des Symbols, die weiß dargestellt sind, werden im Shapefenster gelöscht.
Z: Das Symbol wird nur angezeigt (EOR) und nach dem nächsten Tastendruck wieder gelöscht.
R: Die Richtung des Symbols wählen (0...270 Grad in 90-Grad-Schritten). Bei 90 und 270 Grad wird das Symbol gestaucht.
V: Vergrößerung des Symbols wählen.
↑**Q**: Das Programm wird beendet. Der Tabellenzeiger wird vom Menü wieder auf den Tabellenbeginn gesetzt.
↑**D**: DOS

6.4. ZEICHENSATZEDITOR

Der Zeichensatzeditor erlaubt das Erstellen und Ändern der Zeichensätze. Ein Zeichen hat eine Breite von 7 und eine Höhe von 8 Punkten. Dieses Format wurde gewählt, um eine 80-Zeichendarstellung zu ermöglichen. Die Zeichensätze sind Apple-Toolkit-kompatibel. Ein Zeichensatz wird mit dem Suffix „.SET" abgespeichert.

6.4.1. Display

In der linken, oberen Ecke wird das Zeichen vergrößert abgebildet, daneben in Orginalgröße. Die Tastaturfunktionen und die Angaben der Cursorposition befinden sich an gleicher Stelle wie beim BLOCKEDITOR.

6.4.2. Befehlsübersicht

Die Cursorsteuerung und die Tastaturfunktion ist identisch mit dem BLOCKEDITOR. Die Zeichenwahl wird jedoch mit den Pfeiltasten links/rechts ermöglicht.

6.5. KONVERTER

Der Konverter erlaubt das Übertragen von HGR- oder DHGR-Bildausschnitten in den BLOCK- oder ZEICHENSATZEDITOR. So ist es möglich, Bestandteile von allen handelsüblichen Grafikprogrammen zu verwenden. HGR-Bilder können auf die doppelte Breite des DHGR-Bildschirmes vergrößert oder im normalen Format (gestaucht) übertragen werden. Beim Laden eines Bildes werden Block und Zeichenspeicher überschrieben. So müssen Sie darauf achten, daß Sie schon übertragenene Teile mit den entsprechenden Editoren vorher abspeichern.

6.5.1. Display

Die Anzeige und der Bildschirmaufbau unterscheiden sich von den anderen Programmteilen: Die Überschrift und die gewohnten Kurzanleitungen können nicht dauernd auf dem Bildschirm belassen werden. Das Startmenü erhalten Sie nach Drücken der Return-Taste, die Untermenüs durch den Button 1. Bei „Ausschnitt in

Zeichen konvertieren" wird in der linken unteren Ecke der ASCII-Code des Zeichens angezeigt, das konvertiert werden soll.

6.5.2. Befehlsübersicht Startmenü

1: Normales HGR-Bild laden. Das Bild wird erst nach Übertragung (Taste 2 oder 3) angezeigt.
2: Das HGR-Bild wird in die linke Hälfte des DBLH-Bildschirmes übertragen und angezeigt.
3: Das HGR-Bild wird horizontal auf doppeltes Format vergrößert (Kreise werden wieder rund), übertragen und angezeigt.
4: Untermenü 1 wird aufgerufen und bis zum nächsten Tastendruck angezeigt.
5: Untermenü 2 wird angezeigt.
6: DBLH-Bilder laden und anzeigen.
7: DBLH-Bilder (z.B. konvertierte HGR-Bilder) abspeichern.

6.5.3. Befehlsübersicht Untermenü 1
(Nach Block konvertieren)

Die Befehle werden erst nach Tastendruck (Menü verschwindet) gültig. Der Cursor wird mit dem Joystick bewegt, mit Button 0 wird die Schrittweite versiebenfacht. Button 1 zeigt bis zum nächsten Tastendruck das Untermenü 1.
0...9: Der Bildschirm innerhalb des quadratischen Cursors wird in den Blockspeicher übertragen.
↑M (RET): Sie gelangen ins Hauptmenü zurück.

6.5.4. Befehlsübersicht Untermenü 2
(Zeichen konvertieren)

Die Cursorbewegung erfolgt mit dem Joystick wie bei Untermenü 1. Die Zeichenwahl erfolgt mit den Pfeiltasten, die Übernahme mit SPACE.

6.5.5. Das Maschinenprogramm CONVERT

Dieses Unterprogramm dient zum Übertragen eines HGR-Bildes in das BLOCK-FORMAT mit normal aufeinanderfolgenden Zeilen. Übertragen wird der Bereich $4000-$5FFF (HGR2) nach $6000-$7FFF. Das Programm kann an einer beliebigen Adresse geladen werden und wird durch Aufruf der Startadresse aktiviert.

6.6. STARTPROGRAMM

Dieses Unterprogramm verknüpft ein BASIC-Programm mit dem DBLHTR. Es wird ein EXEC-File erzeugt, der sämtliche Ladefunktionen automatisch durchführt und anschließend das BASIC-Programm startet. Der File wird auf dem letzten aktiven Laufwerk gespeichert.

Auf die Fragen des Programms müssen folgende Antworten gegeben werden:
SOLL DAS PROGRAMM ÜBER ...: Bei längeren BASIC-Programmen muß das Programm hinter die HGR-Seite verlagert werden, da sonst zuerst Bild- und danach BASIC-Störungen auftreten.
PROGRAMMNAME: Name des BASIC-Programmes. Der erstellte EXEC-File wird nach Abschluß aller Angaben mit dem Programmnamen und dem Suffix „.START" gespeichert.
ZEICHENSATZ 1: Name des Hauptzeichensatzes. Diese Angabe kann weggelassen werden, sofern beim Programmstart sichergestellt ist, daß sich ein Zeichensatz im Speicher befindet.
ZEICHENSATZ 2: s. Zeichensatz 1
NAME DER SHAPETABELLE: Filename der benötigten Tabelle. Wird keine Angabe gemacht, kann eine im Speicher befindliche Tabelle verwendet werden.
NAME DER BLOCKTABELLE: Es kann eine Blocktabelle (z.B. zum Titelaufbau) ins MAIN-RAM geladen werden. Wird hier ein Filename eingegeben, wird nach der Ladeadresse verlangt. Diese sollte sich im vom BASIC-Programm freien Speicher befinden.
SOLL DER GRAFIKTREIBER...: Ist der Grafiktreiber vor dem Programmstart schon im Speicher, kann diese Frage mit „N" beantwortet werden. Wenn Sie alle Angaben zu Ihrer Zufriedenheit gemacht haben, können Sie den EXEC-File durch Drücken der Taste „S" abspeichern lassen. Die Taste „E" verläßt das Programm, ohne es abzuspeichern, mit der Taste „N" kann ein weiterer EXEC-File erstellt werden.

7. Adressen und Einsprünge von DBLHTR

7.1. Externe Einsprünge

Diese Einsprünge in DBLHTR können aus der Assemblerebene, d.h. aus Maschinenprogrammen aufgerufen werden.
Vor einem Aufruf muß die Aux-LC aktiviert sein!

Label	Adresse	Funktion	Ein-/Ausgabe
KEYBD	$DØØØ	Zeicheneingabe	Areg
HCOUT	$DØØ3	Zeichenausgabe	Areg
INIT	$DØØ9	Initialisierung der Tabellen und der Grafik	
Blockshapebefehle:			
EXDRAW	$DØØC	Symbol auf Bildschirm	BLK, XLT, XHT, YLT, Xreg
EXMOVE	$DØØF	Symbol zu neuer Position bewegen	BLK, XLT, XHT, YLT, Xreg
EXSCAN	$DØ27	Symbol in Tabelle	BLK, XLT, XHT, YLT, BR, TI
EXCOPY	$DØ33	Tabelle kopieren	Yreg, Xreg
EXCOU	$DØ3C	oben/unten vertauscht	Yreg, Xreg
EXCLR	$DØ39	links/rechts vertauscht	Yreg, Xreg
EXOVL	$DØ36	Überschneidung?	Yreg, Xreg Areg>Ø
Punktbefehle:			
HPLOT	$DØ12	Punkt zeichnen	Areg, Yreg, Xreg, PMODE
HPLOTO	$DØ15	Linie zeichnen	Areg, Yreg, Xreg, PMODE
GETSET	$DØ24	Punkt gesetzt?	Areg, Yreg, Xreg Areg>1
Fensterbefehle:			
SHOO	$DØ18	Fensterinhalt nach oben	–
SHOU	$DØ1B	Fensterinhalt nach unten	–
SHOL	$DØ21	Fensterinhalt nach links	–
SHOR	$DØ1E	Fensterinhalt nach rechts	–
INVER	$DØ2A	Fenster invertieren	–
ESHOW	$DØ2D	Fenster anzeigen	Xreg
EXSWIND	$DØ4B	Fenster speichern	Xreg, vorher EHSHOW!
EXRWIND	$DØ4E	Fenster wiederherstellen	Xreg, vorher EHSHOW!
CLEAR	$DØ5A	Fenster löschen	–
Vergrößerungsbefehle:			
EXCXAGN	$DØ42	Zeichen vergrößern	Areg, XLT, XHT, YLT, TMP1, TMP2, PMODE, RIC
EXDMAGN	$DØ45	Symbol vergrößern	BLK, XLT, XHT, YLT, TMP1, TMP2, PMODE, RIC
MAGN	$DØ48	Block vergrößern	BASE, BASE+1, TZEILE, TI, XLT, XHT, YLT, TMP1, TMP2, PMODE, RIC
Sonstige Befehle:			
EXCPOS	$DØ3Ø	Cursorpositionierung	Xreg, $5Ø
PDL	$DØ3F	Paddleabfrage	BASE – BASE+3 Wert: $FF,Ø,1
EXPEEK	$DØ51	Peek im AUX-Memory	$5Ø,$51 Areg
EXPOKE	$DØ54	Poke im AUX-Memory	$5Ø,$51 Areg
GREY	$DØ57	Schirm grau löschen	–

7.2. Zeropagebenutzung

Im MAIN-Memory werden nur die Speicherzellen $50/51 zur Parameterübergabe verwendet.

```
AUX-Memory:
GL      $5Ø    Parameterübergabe
GH      $51
BLK     $52    Block, Seite
CMD     $53    Befehlsauswertung
COUNT   $54    Zwischenspeicher
TZEILE  $55
TLINKS  $56    Zwischenspeicher für
               linken Rand
YREG    $57    Zwischenspeicher,
               Tiefenübergabe
XREG    $58    Zwischenspeicher
IFLAG   $59    Zeichenausgabemaske
ADD1    $5A    Zwischenspeicher
PMODE   $5B    Plotmode (Ø...2)
TMP1    $5C    Zwischenspeicher
TMP2    $5D
TMP3    $5E
TMP4    $5F
BR      $6Ø    Blockbreite
TI      $61    Blocktiefe
YLT     $62    Blockpos. vertikal (Ø..FF)
XLT     $63    Blockpos. horiz. (Ø..2FF)
XHT     $64
RIC     $65    Richtung bei
               Vergrößerung (Ø..3)
OBEN    $66    Aktuelles Fenster
UNTEN   $67
LINKS   $68
RECHTS  $69
ZEILE   $6A    Textausgabe
SPALTE  $6B
BASE    $6C    Zwischenspeicher
        $6D
HPOS    $6E
        $6F
        $DØ    Zwischenspeicher für
       -$D5    HPLOT TO
        $EØ    Zwischenspeicher für
       -$E2    letzte Plotposition
```

7.3. Wichtige Unterprogramme

(s. auch 7.1.) Die Adressen der Unterprogramme können sich bei Programmänderungen verschieben, sie gelten also nur als Anhaltspunkt.

```
Label   Adr.   Funktion
CHKCOM  $DØ5D  Test auf Komma im Prg.text
GETBYT  $DØ6C  Holt 1 Byte Integer aus
               Programmtext
GETNUM  $DØ76  Holt 2 Byte Integer,
               CHKCOM, GETBYT
GIVAYF  $DØ8Ø  Wandelt 2 Byte Integer in
               Floatingpointformat um;
               Integer muß in A- und
               Y-Register sein.
MOVMF   $DØ8A  Speichert Floatingpointzahl
               in Variable
PTRGET  $DØ94  Holt Variablenspeicher aus
               Programmtext
(Beim Aufruf der obenstehenden Routinen
wird das Applesoft-ROM aktiviert.)
AMPER   $D13B  Befehlsauswertung
```

Befehlsauswertung: Der erste Buchstabe hinter „&“ wird mit der Befehlstabelle CMDTBL verglichen. CMDTBL kann bis zu 254 Buchstaben enthalten. Der letzte Tabelleneintrag muß 0 sein. Wird das gesuchte Zeichen gefunden, wird die zugehörige Adresse aus der Tabelle AMPTBL angesprungen, sonst erfolgt der Rücksprung zum Aufrufer.

7.4. BASIC-Einsprünge der verschiedenen Routinen

Die notwendigen Parameter werden aus dem Programmtext geholt, danach werden die benötigten Speicherbänke aktiviert und der Befehl wird ausgeführt. Nach der Ausführung muß der LC-Bereich des AUX-Memory noch aktiv sein, während der restliche Speicher auf MAIN geschaltet sein muß.

```
Label   Adr.   Funktion
DRAW    $D1A5  Zeichnet Shape
MOVE    $D1DC  Bewegt Shape
SETXY   $D1EF  Setzt Hires-Cursor
PLOT    $D1F9  Zeichnet Punkt
PLOTTO  $D2ØD  Zeichnet Linie vom Hires-
               Cursor zur neuen Position
PSET    $D241  Testet, ob Punkt gesetzt
FLOAT   $D247  Setzt AREG auf Ø, überträgt
               YREG in Variable. Vor Va-
               riablen muß Komma stehen!
FLOAT1  $D24E  Überträgt 2 Byte Integer
               in Variable
SCAN    $D258  Speichert Bildschirmaus-
               schnitt in Shapetabelle
INVERT  $D27B  Invert. aktuelles Fenster
SHOWWND $D27E  Zeigt Fenster an
SETWND  $D29E  Fenstergröße in Tabelle
CURSPOS $D2CA  Positioniert Textcursor
               (wenn im Fenster)
COPY    $D2E2  Kopiert Zeiger v. 2 Shapes
OVL     $D2F7  Überschneidung v. 2 Shapes
GETSHP  $D3ØC  Überträgt Shapepointer in
               Variable
SETSHP  $D315  Überträgt Variable in
               Shapepointer
PUTM    $D323  Überträgt Speicherbereich
               mit EOR auf Bildschirm;
               keine Bankumschaltung!
GETA    $D32A  Umkehrung von PUTM
CLR     $D341  Vertauscht links/rechts,
               legt neue Shapetabelle an
               (nur Offset Ø)
COU     $D356  Vertauscht oben mit unten,
               sonst wie CLR
PADDLE  $D36B  "Digitale" Paddleabfrage
               aller 4 Paddleeingänge
BUTTON  $D38D  Buttonabfrage aller 3
               Buttoneingänge
CMAGN   $D3AE  Vergrößerung eines Zeichens
               (MSB gesetzt!)
DMAGN   $D3FA  Vergrößerung eines Shapes
MMAGN   $D448  Vergrößerung eines
               Speicherbereiches
SWIND   $D48C  Speichert Fenster im Shape-
               bereich, setzt Spointer
RWIND   $D4C6  Stellt Fenster wieder her,
               löscht Spointer
WRECH   $D53D  Rechnet Fenster in Shape um
PEEK    $D553  Peek im kompletten AUX-Mem
POKE    $D565  Poke im kompletten AUX-Mem
GREY    $D598  Löscht Bildschirm
```

7.5. Unterprogramme

Beim Aufruf dieser Programme müssen die benötigten Speicherbereiche aktiviert sein.

```
Label   Adr.   Funktion
DISPGRAF$D5C7  DBLH anzeigen
CLEAR   $D5DA  Fenster löschen
INVER   $D611  Fenster invertieren
CHAR    $D64E  Adr. des auszugebenden
               Zeichens berechnen. In
               $D662 und $D668 (CNUML+1/
               CNUMH+1) muß die Basis-
               adresse des Zeichensatzes
               stehen
HCOUT   $D66C  Zeichenausgabe
CTRL    $D6F6  CTRL-Zeichen-Auswertung
SCRU    $D7D8  Bildschirm im Fenster
               nach oben scrollen
SCRD    $D825  Bildschirm im Fenster
               nach unten scrollen
HOME    $D876  Fenster löschen
CLEARL  $D88Ø  Zeile komplett löschen
CLEAREOL$D89A  Zeile ab Cursorpos. löschen
CLEOS   $D8A8  Fenster ab Cursorpos. löschen
INVERSE $D8CØ  Zeichenausgabe auf INVERSE
NORMAL  $D8C5  Zeichenausgabe auf NORMAL
CDOWN   $D8C9  Cursor eine Zeile tiefer
CURSUP  $D8D8  Cursor eine Zeile höher
KEYBD   $D8E4  Zeicheneingabe über Tasta-
               tur, Zeichen unter Cursor
               wird gelöscht, ESC und ->
               wird verhindert, da Kopier-
               funktion nicht möglich
GETSET  $D9ØC  Punkt gesetzt?
HPLOT   $D922  Punkt setzen
HPOSN   $D964  Berechnung der Zeilenadr.,
               Berechnung des horizontalen
               Bytes und Bits
HPLOTO  $D9AE  Modifizierte ROM-Routine
               zum Zeichnen einer Linie
GETBADR $DA3D  Shapeadresse aus Shapenr.
               berechnen, Shapegröße in
               entsprechende Zwischen-
               speicher eintragen
EORDRAW $DA7D  Shape zeichnen mit
               EOR-Funktion
ORDRAW  $DB13  Shape zeichnen
               mit OR-Funktion
ERADRAW $DB89  Shape zeichnen
BLADRAW $DC3B  Shape schwarz zeichnen
SCANBLK $DCD5  Bildschirmausschnitt in
               Shape umwandeln
SSETUP  $DD77  Fenstergröße in Zwischen-
               speicher ablegen, Vorbe-
               reitung für Shiftroutinen
SHOR    $DDA4  Fenster 1 Bit nach rechts
SHOL    $DDE5  Fenster 1 Bit nach links
SHOO    $DE3A  Fenster 1 Bit nach oben
SHOU    $DE8E  Fenster 1 Bit nach unten
MOV     $DEFE  Shape bewegen; ist neue Pos.
               = alte Pos. -> Rücksprung
BCOPY   $DF3F  Kopiert Shapetabelle
OVLAP   $DF71  Überschneidung v. 2 Blöcken?
SGETADR $DFCE  Vorbereitung Shapeumformung
COPLR   $EØ1F  Links/rechts vertauschen
COPOU   $EØ6F  Oben/unten vertauschen
PDL     $EØCB  Paddleabfrage
MAGN    $E118  Vergrößerung, Auswahl von
               MØAGN - M3AGN entsprechend
               der Richtung
MØAGN   $E13Ø  Vergröß. um Ø Grad gedreht
M1AGN   $E17F  Vergröß. um 9Ø Grad gedreht
M2AGN   $E1C8  Vergröß. um 18Ø Grad gedreht
M3AGN   $E217  Vergröß. um 27Ø Grad gedreht
PSQ     $E26Ø  Zeichnet Quadrat in der durch
               die Vergröß. best. Größe
```

7.6. Tabellen zum Grafiktreiber

```
OB      $E298  Rand oben für Fenster
               in Punkten
UNT     $E2A8  Rand unten (Rand + 1) in
               Punkten
LI      $E2B8  Rand links in Bytes
RE      $E2C8  Rand rechts (Rand + 1) in
               Bytes
SPOINTER$E2D8  Aktuelle Adresse, in der
               nächstes Shape od. Fenster
               gespeichert wird
HTABLEL $E2DA  Low Byte der Zeilenadresse
HTABLEH $E39A  High Byte der Zeilenadr.
XOFFL   $E45A  Spalte aus horiz. Wert
XOFFLR  $E579  Bit aus horiz. Wert
POINT   $E698  Punkt ensprechend XOFFLR
EXTERN  $E69F  Ende des Programms, Adr.
               für Programmerweiterung
```

7.7. Blockshapetabellen

```
SHAPEL  $4ØØØ  Tabelle für Shapeadressen-
               speicher mit Offset Ø
BLOCKADR$4Ø8Ø  Shapeadressenspeicher für
               7 Offsets pro Shape
XL      $44ØØ  Letzte Position, an der
               Shape ausgegeben wurde
XH      $444Ø
YL      $448Ø
TIEFE   $44CØ  Tiefe der Shapes
BREITE  $45ØØ  Breite der Shapes
FREE    $454Ø  Ab hier können Shapes
       -$BFFF  gespeichert werden
```

MultiRam-CX-Karte

16-BIT-65C816 für den Apple IIc

von Lothar J. Dudek

1. Allgemeines

MultiRam-CX ist eine Speichererweiterungskarte für den Apple IIc, die sowohl eine Vergrößerung des Apple-IIc-RAM-Speichers als auch die Verwendung des GTE 16-Bit-Prozessors 65C816 (der ja auch im Apple IIgs steckt) im Apple IIc ermöglicht.
Hersteller der Karte ist die Checkmate Technology Inc., USA. In Deutschland wird die MultiRam-CX von verschiedenen Händlern angeboten.
Der Anwendungsbereich der Karte ist in erster Linie als schnelle RAM-Disk zu sehen, es ist aber auch die Vergrößerung der Appleworks-Schreibtisch-Kapazität und die Verwendung von 16-Bit-Programmen für den 65C816 möglich.

2. Lieferumfang

Die Karte ist in mehreren Versionen erhältlich (256K, 512K, ohne/mit 65C816-Option). Da für alle ICs Sockel eingesetzt sind, kann man kleinere Ausbaustufen durch Einstecken zusätzlicher RAM-Chips oder durch den Tausch der 65C02-CPU gegen den 16-Bit-Upgrade-Kit (65C816-Prozessor und 1 PAL) auch selbst erweitern. Die hier beschriebene Karte enthielt bereits bei Lieferung die volle RAM-Bestückung mit 512K sowie den 65C816-Kit. Zum Lieferumfang gehören außerdem einige mechanische Bauteile und eine doppelseitig beschriebene Diskette.
Die mitgelieferte Dokumentation ist sehr ausführlich und besteht aus je einem Buch zur Hard- und Software der Karte (ca. 50 bzw. 188 Seiten) sowie einem Handbuch über den 65C816-Kit (ca. 80 Seiten).

Der Hersteller gibt auf die Karte eine Garantie von fünf Jahren. Zumindest in den USA wird außerdem über die Händler ein kostenloser Software-Update-Service geboten. Dort ist auch ein Telefon-Service eingerichtet, der in schwierigen Fällen direkt weiterhelfen soll.
Die „Pascal-RAM-Disk" ist eine zusätzliche Option. Zum direkten Vergleich mit den RAM-Disks unter DOS und ProDOS habe ich sie mit in diesen Bericht aufgenommen. Sie besteht aus einer Diskette mit mehreren ATTACH-FILES zur Erweiterung des Apple-Pascal-Betriebssystems und einem Konfigurierungsprogramm. Die Beschreibung (ca. drei DIN-A4-Seiten) befindet sich als Pascal-Textfile auf der Diskette.

3. Einbau

Zum Einbau der Karte muß der Apple IIc geöffnet und die Tastatur nach oben geklappt werden. Die CPU und MMU müssen aus ihren Fassungen gezogen werden. Ein Ausbau des Disk-Laufwerkes ist nicht erforderlich. Der MMU-Chip kommt in eine Fassung auf der MultiRam-CX-Karte, die CPU wird nicht mehr benötigt, da sie durch den 65C816 ersetzt wird. Danach wird die Karte mit vier an ihrer Unterseite befindlichen Stiftkontaktreihen in die freien Fassungen von CPU und MMU gesteckt. Das ist alles.
Die Karte ist so ausgelegt, daß sie vollständig unter der Tastatur des Apple IIc Platz findet. Ein Versteifungsbügel, der sich ursprünglich darunter befindet, muß entfernt werden. Seine Aufgabe übernehmen sechs Stützen auf der MultiRam-CX-Karte.
Vor dem Schließen des Gehäuses wird empfohlen, das mitgelieferte Diagnoseprogramm zu starten, um sicher zu gehen, daß der Umbau erfolgreich war und der Apple IIc wieder zusammengesetzt werden kann.
Die Installation ist dank der gut illustrierten Anleitung problemlos durchzuführen und dauert mit dem Test weniger als eine Stunde. Auch das Aufhebeln des Apple-IIc-Gehäuses (s. Erfahrungsbericht v. U. Tönnies im Peeker 6/86), das u.U. unschöne Folgen nach sich zieht, ist nach der mitgelieferten Anleitung vermeidbar.

4. Integration

Die Karte ist ausschließlich mit CMOS-Chips bestückt. Dies entspricht dem Stand der Technik, dürfte aber wohl auch erforderlich sein, denn die Karte stellt für das Netzteil des Apple IIc eine zusätzliche Belastung dar (1,2 Watt), die so in zumutbaren Grenzen gehalten wird.
Der Speicherbereich der MultiRam-CX-Karte ist im 62C02-Mode (dieser kann bekanntlich vom 65C816 emuliert werden) in Form von acht 64K-Blöcken (Banks) organisiert. Die Auswahl der einzelnen Banks erfolgt durch Softswitches im Bereich $C07X.
Im 16-Bit-Modus (Native Mode) des 65C816 kann der gesamte 512K-Bereich linear adressiert werden, darüber hinaus sind nach Angaben des Handbuches maximal bis zu 8 Megabyte möglich.

5. Kompatibilität

Die Karte und die mitgelieferte Software sind laut Manual zu allen Standard-Apple-Betriebssystemen sowie zu ProntoDOS, Diversi-DOS und Rana-DOS kompatibel.
Nicht kompatibel sind dagegen David-DOS sowie alle Programme, welche DOS oder ProDOS in die Auxiliary-Bank des Speichers oder DOS vollständig in die Language-Card verschieben. Weitere Einschränkungen bestehen bei allen kopiergeschützten Programmen, die sich – zumindest mit den mitgelieferten Utilities – nicht in die RAM-Disk kopieren lassen. Bei meinen Versuchen ließen sich beispielsweise „Multiplan" unter DOS oder „Mailmerger für Appleworks" unter ProDOS nicht in die RAM-Disk übertragen.
Kalkulationsprogramme wie Supercalc 3a, Magicalc und Flashcalc sollen angeblich ebenfalls kompatibel sein. Darüber hinaus sollen sie den größeren verfügbaren Speicher erkennen und für größere Spreadsheets nutzen.

Für die Hardware-Kompatibilität werden keine Einschränkungen gemacht. Das ist besonders wichtig für den Anschluß eines 800K-Uni-Disk-Laufwerks, denn der Dump des MultiRam-CX kann bis zu vier 5,25"-Standard-Apple-Disketten erfordern.

6. Software

Auf der MultiRam-CX-Karte selbst befindet sich keinerlei Software, so daß je nach Anwendung immer eines der mitgelieferten Boot- und Kopierprogramme nach dem Laden eines Betriebssystems gestartet werden muß.

Läßt man die Textdateien für Informationen über letzte Änderungen außer acht, so enthält die Begleitdiskette auf zwei Seiten etwa 216K Programme für Appleworks und ProDOS und etwa 12K auf gesondert formatierten Spuren für DOS. Die Pascal-Option enthält noch einmal ca. 25K Programmfiles.
Wegen der Fülle der Programme sind hier nur die wesentlichen Merkmale aufgeführt:

– RGB-Demo (in DHGR-Farbgrafik)
– Appleworks-Extender zur Erweiterung des Appleworks-Schreibtisches auf bis zu 413K, SAVE-Routinen für Appleworks-Files, die größer als 140K sind, also auf mehreren Disketten ablegt werden müssen
– RAM-Disk-Treiber zur Initialisierung der RAM-Disks
– Utilities zur Konfigurierung der RAM-Disks (Anzahl, Speichertiefe) und zum Kopieren von Dateien in eine RAM-Disk
– Speicher-Testprogramm, welches die einzelnen Speichersektoren (Banks) der MultiRam-CX-Karte auf einem HGR-Bildschirm darstellt und den gerade getesten Bereich mit Gut/Schlecht-Symbolen markiert. Bei Fehlern lassen sich damit auch die defekten Chips direkt lokalisieren.

Neben diesen spezifischen Programmen erwirbt man auch ProDOS 1.1.1, BASIC.SYSTEM und den ProDOS-Filer, so daß man nach dem Booten der Begleitdiskette gleich die erforderlichen ProDOS-Arbeitsdisketten herstellen kann. Zum Anlegen von Auto-Boot-Disketten für DOS wird eine DOS-Master-Disk und das FID benötigt. (Gleiches gilt sinngemäß für Pascal.)

7. Ergebnisse

7.1. RAM-Disks

Für die Konfigurierung von RAM-Disks unter DOS 3.3, ProDOS und Pascal greift man auf die mitgelieferte (oder optionale) Software zurück, mit der man unter DOS 3.3 und unter Apple-Pascal 1.1 bzw. 1.2 bis zu zwei RAM-Disks (auch unterschiedlicher Größe) installieren kann. Apple-Pascal 1.3 wird noch nicht unterstützt. Unter ProDOS setzt man vorteilhaft eine RAM-Disk mit Subdirectories ein. Für alle Betriebssysteme können Auto-Boot-Disketten angelegt werden, die sich oder weitere Disketten vollständig in die RAM-Disk kopieren.

Besonders erwähnenswert ist in diesem Zusammenhang die Möglichkeit, zwischen den Betriebssystemen DOS und ProDOS (nicht Pascal) hin- und herzuschalten. Einmal konfigurierte RAM-Disks bleiben bis zum Ausschalten des Computers intakt (s.u.) und können nach (Warm-)Booten des jeweiligen Betriebssystems wieder benutzt werden. Auf Files, die unter einem anderen Betriebssystem gespeichert wurden, kann jedoch nicht zugegriffen werden. Weit weniger gutmütig verhält sich demgegenüber die RAM-Disk unter Pascal. Sie läßt keine weiteren RAM-Disks unter anderen Betriebssystemen zu, und die in ihr gespeicherten Dateien überstehen keine Neukonfigurierung der Disk.

7.1.1. ProDOS

Es stehen sowohl /RAM (wie gewohnt mit 64K) als auch /MRAM mit 512K zur Verfügung. Mehrere verschiedene Systemfiles können sich gleichzeitig in der RAM-Disk befinden und man kann blitzschnell zwischen ihnen mit „-XXXXX.SYSTEM" hin- und herschalten. ProDOS selbst läßt sich zwar in die RAM-Disk übertragen, jedoch nicht von dort starten („relocation configuration error").
Da ProDOS 1.1.1 auf der Begleitdiskette mitgeliefert wird, habe ich ältere ProDOS-Versionen nicht getestet. Die Apple-IIc-ProDOS-Uhr (M2000 von IDW) mit zugehörigem Treiber ist mit der Karte voll funktionsfähig.
Nicht funktionsfähig sind – im Gegensatz zum ProDOS-Filer – die zur Grundausstattung des IIc gehörenden System-Utilities: Sie führen zusammen mit /MRAM einen Systemzusammenbruch herbei. Bei erneutem Warmbooten von ProDOS und Neuinitialisieren der RAM-Disk kann jedoch auf alle zuvor in der RAM-Disk gespeicherten Files wieder zugegriffen werden.

7.1.2. DOS 3.3

Nach Ablauf des RAM-Disk-Installationsprogramms stehen zwei RAM-Disks mit 384K und 192K zur Verfügung. Die 64K auf der Hauptplatine des Apple-II-Aux-Memory werden norma-

Double-Hires-Tools

von Matthias Meyer

Zwei preisgünstige Programmpakete für doppelt-hochauflösende Grafik auf dem Apple IIc und IIe (mit 64K-Karte):

DHGR-Tool für Applesoft

Diskette und Manual, Einführungspreis DM 28,–

Diese Ampersand-Programmsammlung für Double-Hires und -Lores läuft unter Applesoft, und zwar sowohl unter DOS 3.3 als auch unter ProDOS. Unter anderen wurden folgende Befehle implementiert:

&1 und &2 wählen 1. und 2. Zeichensatz,
&CLEAR löscht die DHGR-Seite,
&COLOR= und &HCOLOR= wählen Double-Lores/Hires-Farben,
&DRAW und &XDRAW zeichnen DHGR-Shapes,
&DRAW AT zeichnet Grafikbeschriftungen (ASCII-Strings),
&GR, &HGR, &H, &TEXT, &T usw. schalten verschiedene Grafik- und Text-Modi ein,
&HLIN und &VLIN plotten waagrechte und senkrechte Double-Lores-Linien,
&HPLOT und &XHPLOT plotten DHGR-Linien,
&SCALE= und &ROT bestimmen Größe und Rotation von Shapes,
&LOAD und &SAVE laden und speichern Grafikseiten,
&HELP zeigt alle Befehle an,
&PRINT: Schnittstelle für Superdump aus Peeker 6/85
und vieles mehr.

DHGR-Tool für Kyan-Pascal

Diskette und Manual, Einführungspreis DM 28,–

Das Kyan-Pascal-Tool umfaßt ähnliche Prozeduren wie die nebenstehenden Ampersand-Routinen, wobei jedoch noch einige Befehle, z. B. Procedure Swaphires, Background, Circle usw., sowie einige Datentypen, z. B. Shape, Chrset usw. zusätzlich aufgenommen worden sind.

Bei dem Kyan-Tool sind die Zeichensätze und die „Lookup"-Tabellen für die sehr schnellen Plotbefehle auf die 64K-Karte gelegt worden, und das Hauptmodul selbst befindet sich in der Bank 2 der Language-Card, ohne Kix-Reboot zu zerstören. Damit eignet sich dieses Kyan-Modul besser als andere Kyan-Grafik-Programme zur Einbindung in eigene Anwendungsprogramme.

Besondere Merkmale beider Utilities:

– Grafikbeschriftungen in acht Richtungen und beliebiger Größe möglich
– Verwaltung zusätzlicher Grafikseiten in der zweiten 64K-RAM-Bank.
– Alle Programmteile und Tabellen residieren außerhalb des BASIC- bzw. Pascal-Arbeitsbereichs.

Hüthig Software Service · Postfach 10 28 69 · 6900 Heidelberg 1

lerweise mit in die RAM-Disk einbezogen. Mit dem mitgelieferten Konfigurierungsprogramm kann man dies allerdings unterdrücken. Die zweite Disk hat dann 128K, um die Verwendung von Programmen zu ermöglichen, welche den AUX-Bereich bereits benutzen.

7.1.3. Pascal

Unter Apple-Pascal (ich verwende Version 1.2) können zwei zusätzliche Disk-Laufwerke (#11 und #12) konfiguriert werden. Dies ist von der zusätzlichen Kapazität her betrachtet ein ziemlicher Gewinn. Im Vergleich zu DOS oder ProDOS ist die Pascal-RAM-Disk allerdings etwas langsamer.
Die Pascal-RAM-Disk kann zum SYSTEM-Volume (#4) deklariert werden, wodurch das ständige Nachladen von FILER, EDITOR etc. um einiges beschleunigt werden kann.

7.1.4. Appleworks

Das mitgelieferte Appleworks-Expansion-Modul (V4.5) soll laut Handbuch nur mit den Appleworks-Versionen 1.2 und 1.3 arbeiten (bei der deutschen Version von Appleworks 1.3 ist offenbar Vorsicht geboten; bei mir ließ sie sich nicht erweitern). Upgrades von älteren Appleworks-Versionen erhält man beim jeweiligen Apple-Händler, evtl. gegen geringe Kosten.
Das Programm-Modul modifiziert die Appleworks-Files auf der Appleworks-Diskette. Dadurch erhält Appleworks folgende zusätzliche Eigenschaften:
– Bis zu 413K Schreibtischkapazität (weniger, wenn zusätzlich noch eine RAM-Disk konfiguriert ist)
– Laden des gesamten Appleworks-Programms inkl. Druckerkonfiguration in den Arbeitsspeicher (kein Nachladen von einzelnen Appleworks-Modulen mehr)
– automatisches Speichern von Dateien mit über 140K Länge auf mehreren Disketten
– Datum und Zeit auf dem Appleworks-Schirm (nur mit angeschlossener ProDOS-Uhr)
– Eingabe von aktuellem Datum und aktueller Zeit in Datenbank-Felder auf Tastendruck
– Automatisches Einlesen des Datums in die Berichtsformate
MultiRam-CX gefällt mir in Verbindung mit Appleworks am besten. Die Vergrößerung der Schreibtischkapazität und der wesentlich schnellere Zugriff auf die einzelnen Appleworks-Module erhöhen den Nutzen von Appleworks ganz wesentlich.

7.2. 16-Bit-Kit (65C816)

Zur Zeit sind nur wenige 16-Bit-Programme für den 65C816 auf dem Markt, so daß dieser Kit in erster Linie für Assemblerprogrammierer interessant ist. Hierfür enthält das zugehörige Handbuch eine Menge detaillierter Informationen über den Prozessor, die Speicheraufteilung der Karte, grundlegende Programmierhinweise für den 65C816 und eine Übersicht über den Stand an Literatur (1985) und über bereits verfügbare und geplante Programme und Betriebssysteme.
Neben verschiedenen Assemblern (Merlin-Pro V2.4 und höher, ORCA/M 4.0, S-C Macro-Assembler V2.0 und LISA 3.2 Macro-Assembler), welche den 65C816/65C802 unterstützen, finden sich in diesem Handbuch Kurzbeschreibungen und Herstellerangaben folgender Hochsprachen:
– BASIC-16 (aufwärtskompatibel zu APPLESOFT mit mehreren Erweiterungen)
– Pascal-16 (compiliert in P- und 65C816-Native-Code, Transfermöglichkeit von APPLE-PASCAL 1.1 und 1.2)
– Kyan-Pascal (volle Pascal-Implementation unter ProDOS, compiliert in 65C816-Native-Code)
– C (weiterentwickelte Version des Aztec C für 65C02)
– FORTH (erweitertes FIG-FORTH und FORTH-83)

Alle aufgeführten Hochsprachen sind bei Drucklegung des Handbuches für das Frühjahr 1986 angekündigt und dürften heute teilweise bereits verfügbar sein.
An 16-Bit-Betriebssystemen wird das MAX-OS angekündigt (UNIX-ähnlich mit Dateiverwaltungssubsystem, Command-Interpreter, I/O-Treiber-Subsystem, Filetransfer- und Formatter-Routinen), welches auch multitaskingfähig sein soll. MAX-OS soll Bestandteil des 65C816-Kit sein, ist aber offenbar noch nicht verfügbar. Als Käufer des Kit erwirbt man eine Anwartschaft auf die Nachlieferung von System-Disk und Handbuch.
Inzwischen erschienen ist das seit langem angekündigte Programm 'VIP-Professional', welches eine Mac-ähnliche Benutzeroberfläche besitzt und vollständig LOTUS-1-2-3-kompatibel ist. Erste Erfahrungen mit diesem Programm (Rechenblatt, Datenbank und Grafik), welches mir erst seit kurzem vorliegt, scheinen die vielversprechenden Ankündigungen zu bestätigen. Eine eingehendere Beschreibung würde jedoch den Rahmen dieses Berichtes sprengen; sie wird später erscheinen.

Fazit

Die Karte macht einen guten und professionellen Eindruck. Auch als technischer Laie kann man sich an den Umbau seines Apple IIc wagen, ohne größere Schwierigkeiten befürchten zu müssen.
Während der zweimonatigen Betriebszeit fand bei mir kein einziger Systemzusammenbruch statt, der auf die MultiRam-Karte oder auf die im Zusammenhang damit verwendete Software zurückzuführen gewesen wäre. Auch wenn dies keine endgültige Beurteilung sein kann, ist wohl an der Qualität der Hard- und Software nichts auszusetzen.
Mit 640K-Speicherkapazität und 16-Bit-Prozessor (noch mit Einschränkungen) erscheint der Apple IIc in seinem handlichen Gehäuse auch gegenüber anderen Megabyte-Computern in einem recht günstigen Licht.
Trotz des stolzen Preises (in USA ca. US-$ 299,–, in Deutschland ca. DM 1600,–) erscheint mir diese Investition durchaus lohnend, da man auf ein zweites, externes Disklaufwerk (das ja bei weit geringerer Kapazität durchaus auch seinen Preis hat) in den meisten Fällen verzichten kann.
Der 65C816-Kit bringt dem reinen IIc-Anwender zur Zeit keine Vorteile (auch VIP-Professional läuft nämlich auf dem IIc nur im 8-Bit-Emulation-Mode), ist aber für denjenigen interessant, der eine technisch einigermaßen vernünftige, jedoch billigere Lösung als den IIgs sucht, um den 65C816 kennenzulernen und Programme dafür zu schreiben.

ProDOS-Editor 1.0

Applesoft-Editor unter ProDOS-Betriebssystem

von U. Stiehl

1984, Diskette und Manual, DM 38,–
ISBN 3-7785-1024-X

Mit diesem neuen Editor – übrigens der bislang einzige deutsche ProDOS-Editor – wird dem Applesoft-Programmierer ein Werkzeug zur effektiven Programmierung unter dem Betriebssystem ProDOS gegeben, denn die früheren Editoren sind allesamt unter ProDOS nicht mehr lauffähig.

Unter anderem sind folgende Features implementiert worden:

– Zeilenorientierter Editor mit jedem erdenklichen Redigierkomfort (Insert, Delete, Tab, Restore, freie Cursorbewegung in allen vier Richtungen, Eingabe von Ctrl-Buchstaben in Applesoft-Zeilen usw.)
– Renumber (Zeilen-Umnumerierung)
– Xreference (sortierte Variablenliste)
– Suchen von Tokens, Strings und Variablen
– dezimale und hexadezimale Umrechnungen
– Ausführung von Monitorbefehlen aus dem Editor heraus
– Listen des Applesoft-Programms in speicherinterner Form als Hex-Dump
– Suchen von Hex-Folgen, Adressen oder Speicherstellen im gesamten RAM-Bereich einschließlich der Language-Card
– frei definierbare Tastatur-Macrobefehle

Der Applesoft-Editor liegt in einem von ProDOS geschützten Bereich und läßt sich per Tastendruck vorübergehend abschalten und ebenso einfach wieder aktivieren.

Gerätevoraussetzung: Apple II+, IIe oder IIc, 40 Zeichen/Zeile

Hinweis: Der Applesoft-Editor für DOS 3.3 befindet sich auf Sammeldisk # 16 für Fortsetzungsbezieher. Siehe Peeker 4/86, S. 13.

Hüthig Software Service,
Postfach 10 28 69,
D-6900 Heidelberg

Rechtsschutz von Computer-programmen

von Dipl.-Ing. Hans Raible

1. Patentschutz

Der Patentschutz für Programme ist insoweit ausgeschlossen, als das Programm für gängige Computer bestimmt ist, so z.B. für einen normalen Apple.
Nicht ausgeschlossen ist der Programmschutz, wenn auch am Rechner etwas geändert wird, also z.B. bei einer Kanone, wenn am Feuerleitrechner technisch etwas geändert und ein entsprechendes neues Programm geschrieben wird, wobei die Kombination von Software und geänderter Hardware neue Effekte bringt.

2. Urheberrechtsschutz

Der Vorsitzende Richter am 1. Zivilsenat des Bundesgerichtshofs – dieser ist für Urheberrechtssachen zuständig – machte hierzu bei einem Vortrag in Stuttgart interessante Ausführungen.
Der Schutz eines Programms durch Urheberrecht setzt eine bestimmte „Eigentümlichkeit" voraus, etwa zu vergleichen mit der „Erfindungshöhe" im Patentrecht. Nach Schätzung dieses Richters erfüllt die große Mehrzahl der Programme dieses Kriterium nicht, so daß diese Programme nicht urheberrechtlich geschützt werden können.
Den Programmautoren, denen das Urheberrecht nicht hilft, kann aber über das Gesetz gegen unlauteren Wettbewerb geholfen werden, wenn ihre Werke kopiert oder in ähnlicher Form übernommmen werden. Die Dauer eines solchen Schutzes steht im Ermessen des Gerichts und kann etwa zwischen 6 Monaten und 3 Jahren betragen. Man bekommt daher für die Großzahl der Programme eine Art Minipatent auf der Basis des Richterrechts. Für die Dauer des Schutzes spielt dabei eine Rolle, ob direkt kopiert wurde (dann längerer Schutz) oder ob der Nachahmer Änderungen vorgenommmen hat (dann kürzerer Schutz).

3. Französisches Gutachten

Für die Frage der Eigentümlichkeit eines Programms ist ein Gutachten interessant, das ein französischer Richter geschrieben hat; Anlaß war die Frage, ob bestimmte Spiele der Firma Atari in Frankreich Urheberrechtsschutz genießen. Der Richter sagte: „Ziel ist nicht, ein neues Recht zu schaffen, das unsere gegenwärtige Gesetzgebung nicht zuläßt, sondern den Instanzrichtern zu helfen, eine bemerkenswerte Unordnung zu beenden."
Zu den einzelnen Elementen eines Atari-Spiels sagte er folgendes (zitiert nach GRUR Int. 1986, Seite 460):

a) *Das Spiel.* Originalität vorausgesetzt hindert nichts, eine Spielidee nebst Spielregeln zu schützen, sofern sie nur niedergelegt ist und Form gefunden hat. Geschützt ist dann aber nicht etwa das Spiel selbst, sondern nur die künstlerische Darstellung der Spielelemente.
b) Nach denselben Regeln wie ein Bühnenbild ist auch die gezeichnete *Darstellung des Spielfeldes* geschützt. Die Darstellung der bislang der Rechtssprechung bekanntgewordenen Videospiele beschränken sich jedoch auf elementare Zeichen, deren Originalität sehr fraglich erscheint.
c) Die *Musikbegleitung.* Sie unterfällt den Vorschriften des Gesetzes von 1957 und kann wie in jedem zusammengesetzten Werk als solche geschützt werden. Da es dem Autor eines Spiels jedoch entscheidend auf den Spielzweck ankommt, bedeutet die Verletzung der Musik, selbst wenn jene original sein sollte, nicht zugleich die Verletzung des Videospiels. Etwas anderes könnte nur für ein Videospiel gelten, das hauptsächlich die Musik in Szene setzt.
d) Für die diesem Typ von Spiel eigene *Inszenierung und Bewegung der Handlung* ist Filmbildschutz geltend gemacht worden. In der Sache Williams hat die Cour d'appel dem Videospiel Urheberrechtsschutz jedoch versagt, da „das Wesentliche eines künstlerischen Werkes in seiner Unveränderbarkeit liege". In diesem Fall jedoch greift der Spieler nicht in das Werk selbst ein, vielmehr bleiben die Bewegungen der steuerbaren Objekte innerhalb des vorgegebenen Rahmens, den der Spieler weder überschreiten noch verändern kann. Die Begründung der Vorinstanz ist daher mit dem Schutz, den im Rahmen der Aufzählung auch Filmwerke genießen, unvereinbar.
e) Das *Programm.* Seine Stellung innerhalb der Einzelelemente ist bedeutend, es ist die Seele des Spiels. Vermag es, selbst schutzwürdig, seine eigene Originalität – so sie gegeben ist – dem Videospiel als Ganzes zu verleihen?
Auch für das Spiel als Ganzes bleibt die Originalität das für die Schutzfähigkeit entscheidende Kriterium. Reicht es aber aus, daß ein einziges Element original ist? Zweifellos nicht, da die Summe schutzfähiger Elemente einem an sich banalen Spiel den Schutz nicht verschaffen kann. Mag die Originalität mit der Seltenheit eines Werkes zusammenhängen, so können sowohl der Faktor der Zeit wie der Anzahl ein Werk banal erscheinen lassen. Im übrigen sind die Einzelbestandteile zu gewichten. Einige, wie etwa das Programm, sind bedeutend. Die Auswirkung des einzelnen auf die Originalität des Ganzen ist unterschiedlich. Es handelt sich um eine Tatfrage, und der Gegenstand der Prüfung der Originalität ist das gesamte Spiel selbst. Daß das verwandte Programm – im übrigen häufig ein Speicherelement, das in allen konkurrierenden Spielen eingebaut ist – selbst geschützt sein mag, erlaubt nur in Ausnahmefällen den Schluß, daß ein an sich banales Spiel Urheberrechtsschutz zu genießen vermag. (Zitat Ende)

In der Sache wurden die Urteile der französischen Instanzgerichte, die mit recht fragwürdigen Formulierungen die Frage der Urheberrechtsschutzfähigkeit verneint hatten, aufgehoben und zur neuen Verhandlung zurückverwiesen.
Wegen der hohen Kosten für Gutachten ist ein Streit über Fragen der Schutzfähigkeit nach Urheberrecht nur für reiche Parteien zu führen. Das „Minipatent auf Basis Richterrecht" dürfte dagegen leichter zu erhalten sein und daher derzeit für die Mehrzahl der Programmautoren der einzige erreichbare Schutz sein.

PEEKER Börse

Biete Hardware

68000 Karte + 1 MB für Apple IIe + AP20 + AP26 von IBS + CP/M 68K + Debug + Ass. + C-Compiler (DR). Pr. n. VB HPX-84 DIN A3 Flachbrettplotter NP DM 1200 für DM 800,–.
J. Hofer, Tel. 0821/514762

Apple II + 64KB mit VIDEX- Karte; 2 Disk; Alpha-Key Tastatur; Monitor; 2 Druckerinterf. Preis VHB. Stubbe, Tel. 040/666962

Apple IIe + Mon. + 2 LW + Z80 + 80 Zex. + SSC + Imagewriter + Eprommer + Lit. + Softw. 10 Mon. alt. Preis VS.
Tel. 0231/810054.

Apple IIe, 2 Drives, Monitor, Drucker CPM44+56, Pascal, DOS 3.2. 1+3.3, Quick-File II u.v.a. Prog., Handbü., Z80, 80Z, zu verk., Tel. 07641/412 86

Apple IIe mit Monitor, 2 Disk. Laufwerk, 80Z-Karte, Joystick, UCSD-Pascal, 2-Centronics-Interface, Apple-Writer/+ Lit. für DM 2200,–. M. Imhof, Schafmattstr. 1, CH-8841 Gross, T. 055/534785

Apple Slot Platinen mit Verg. Kontakten supergünstig: DM 23,50/St. bei Elektronikversand Korb, Postf. 1331/6115 Münster 1

Apple-IIe-enhanced + 7811- ARITH. PROC + 2 Disk-II + Kont. + Epson Apl. + Z80 + Taxan IIe-8064 (RGB, 80Z, 64KB) + Taxan RGB-Vision II + TITAN ACCEL 2E + Apple-Parr + Softw. + Lit. DM 4250,–.
Jan Abbink, T. 0202/714292

Apple IIe + Disk II + Mon. + SSC + Imagewriter + Erw. 80Z + 512K-RAM-Disk für Pascal + Pascal+.2 DM 2700,–. Alexander Seggerman, T. 06172/303474

Gewerbliche Anzeigen sind mit G gekennzeichnet.

Biete Software

Erstelle Software nach Wunsch! Info (0,80 DM) bei: W. Rittmeyer, Wehrbruchweg 30, 4060 Viersen 1. G

* **DISKETTEN** *
* 51/4" 48 tpi, DM 0,99, 2D *
* 31/2" 135 tpi, DM 3,–, 1DD *
* 31/2" 135 tpi, DM 3,50, 2DD *
* 3" Markendisk. DM 7,20 *
* Allg. Austro-Agent. Ringstr. 10 *
* D-8057 Eching, T.08133/6116 *
G

ÖKOLOPOLY für Apple II Disk DM 30,–. Info gratis. Tel. 0921/566 11 ab 17 Uhr.

Viel Software zu fairen Preisen: Über 100 professionelle Programme aus verschiedenen Kategorien und Softwarehäusern. Info (0,80 DM) bei: W. Rittmeyer, Wehrbruchweg 30, 4060 Viersen 1. G

Apple II: DFÜ-Kermit, Pascal satt, je Disk DM 15,–. Public Domain, Schulpro. u. a. Gratisinfo: Fa. W. Muhle, Waldwinkel 3, 2105 Seevetal 3 G

Finanzbuchhaltung. Frei wählbarer Kontenrahmen, Summen- und Saldenliste, Journal, Kontoauszüge autom. UST-Verbuchung ... DM 300,– (II+) BM-Software, Poetenweg 44, 4800 Bielefeld 1 G

Topsoftware für Apple II: Sporttabellenverwaltung: nur 40,– PFS-Programme: 300,–, BALLblazer: 100,–, Superbase: 300,–. Info (0,80 DM) bei: W. Rittmeyer, Wehrbruchweg 30, 4060 Viersen 1 G

TURBO Pascal Turtlegrafik-Paket für Apple II, kennt alle Befehle der UCSD-Unit, DM 89,– von Fa. Jochen Tucht, Rudolf-Albrecht-Straße 52, 3052 Bad Nenndorf. G

Profi. CAD-Prog. CASCADE I für Apple IIe preisgünstig abzugeben. Preis VH. Tel. 0420 3/21 60 n. 18.00 Uhr.

Verschiedenes

APPLE REPARATUREN
(auch compatible M-boards, z.B. **Atlas, Arca, CES, Datastar, Dipa, Lasar, Mewa, PC-48 + 64, Plato, Radix, o. ae.) sowie Zusatzkarten und Disk-Drives** führt unser Spezialistenteam mit mehr als 5-jähriger Kunden- und Reparatur-Dienst-Erfahrung, garantiert zuverlässig und besonders kostengünstig aus.
Bitte genaue Fehlerangabe sowie Tel. Nr. für evtl. Rückfragen nicht vergessen.
Auf Wunsch Kostenvoranschlag.
aaa-electronic gmbh
Habsburgerstr. 134, 7800 Freiburg, Tel. 0761/276864, Tx. 772642aaad G

Suche Peeker 1/84-9/86, Sammeldisk 1-19, Zeichensätze für DMP-Charger, Tel. 0202/522402 nach 18.00 Uhr.

Welche Lehrer (Gymnas.) sind am Tausch von Unterrichtsmaterial zu Biologie, Chemie, Physik im AppleWorks-Format (oder AWriter) interessiert? Chiffre-Nr. **P1009**

Katalogerstellung mit Macintosh; wer hilft? Angebot an: Peter Sonntag, Dorfstr. 16, 7832 Sasbach, Tel. 07662/1453 G

Wieder allein

Schade, mit dem PEEKER verlieren wir Apple User eine wertvolle Informationsquelle.

Die A.U.G.E. macht weiter. Seit 9 Jahren Treffpunkt für Apple User mit:
55 Regionalgruppen, 20 Arbeitsgemeinschaften, 3 Datex-P und 12 lokale Mailboxen, jede Menge Software für II+, IIe, IIc, IIgs, Mac, ST, XT, AT und dem USER MAGAZIN.

Kontakt: 0208 / 67 51 41

A.U.G.E. e.V.
Postfach 11 01 69
D-4200 Oberhausen 11

Apple User Group Europe e.V. - der größte Computerverein im deutschsprachigem Raum

Für Ihre Unterlagen

Abonnement bestellt

am: _____

Vertrauensgarantie:
Ich habe davon Kenntnis genommen, daß ich die Bestellung schriftlich durch Mitteilung an den Dr. Alfred Hüthig Verlag GmbH, Postfach 10 28 69, 6900 Heidelberg innerhalb von 7 Tagen widerrufen kann. Zur Fristwahrung genügt die rechtzeitige Absendung des Widerrufs (Datum des Poststempels).

Peeker
Leserservice
Postfach 10 28 69
6900 Heidelberg

peeker

Abo-Karte

Ja, ich möchte **Peeker** abonnieren.
Liefern Sie mir **Peeker** ab Ausgabe . . . . . zum Jahresbezugspreis von z.Zt. DM 75,– (Inland) inkl. MwSt. Die Lieferung erfolgt frei Haus. Porto, Verpackung und Zustellgebühren übernimmt der Verlag. Der Jahresbezugspreis für das Ausland beträgt z.Zt. DM 75,– plus DM 20,– Versandspesen.

X _____
Datum 1. Unterschrift

Bitte lesen!

Vertrauensgarantie: Ich habe davon Kenntnis genommen, daß ich die Bestellung schriftlich durch Mitteilung an den Dr. Alfred Hüthig Verlag GmbH, Postfach 10 28 69, 6900 Heidelberg innerhalb von 7 Tagen widerrufen kann. Zur Fristwahrung genügt die rechtzeitige Absendung des Widerrufs (Datum des Poststempels).

X _____
Datum 2. Unterschrift

Verlagshinweis: Das Abonnement verlängert sich zu den jeweils gültigen Bedingungen um ein Jahr, wenn es nicht 2 Monate vor Jahresende schriftlich gekündigt wird.

Wir können nur Bestellungen mit zwei Unterschriften bearbeiten.

Für Ihre Unterlagen

Folgende Bücher bestellt:

_____
_____
_____
_____
_____

am: _____

bei:

Peeker
Versandbuchhandlung
Postfach 10 28 69
6900 Heidelberg 1

peeker

Buch-Karte

Bitte senden Sie mir gegen Rechnung folgende Bücher:

- ☐ Bühler, Applesoft-BASIC, 3-7785-1094-0, DM 38,–
- ☐ Eggerich, dBase II, Bd. 1, 3-7785-1147-5, DM 39,80
- ☐ Eggerich, dBase II, Bd. 2, 3-7785-0987-X, DM 39,80
- ☐ Eggerich, dBase II, Bd. 3, 3-7785-0988-8, DM 39,80
- ☐ Gabriel, Applewriter, 3-7785-1234-X, DM 35,–
- ☐ Hagenmüller, Microsoft-BASIC, Bd. 1, 3-7785-1038-X, DM 38,–
- ☐ Juhnke/Redlin, Apple Pascal, Bd. 1, 3-7785-1246-3, DM 42,–
- ☐ Kehrel, Apple Assembler lernen, Bd. 1, 3-7785-1151-3, DM 38,–
- ☐ Kehrel, Apple Assembler lernen, Bd. 2, 3-7785-1170-X, DM 38,–
- ☐ Schäpers, ProDOS Analyse, 3-7785-1134-3, DM 68,–
- ☐ Schäpers, Bewegte Apple-Graphik, 3-7785-1150-5, DM 58,–
- ☐ Stiehl, Apple DOS 3.3, 3-7785-1297-8, DM 28,–
- ☐ Stiehl, Apple ProDOS, Bd. 1, 3-7785-1098-3, DM 28,–
- ☐ Stiehl, Apple ProDOS, Bd. 2, 3-7785-1036-3, DM 30,–
- ☐ Stiehl, Apple Assembler, 3-7785-1047-9, DM 34,–
- ☐ Wassermann, Apple IIc Handbuch, 3-7785-1157-2, DM 35,–

Datum Unterschrift

Für Ihre Unterlagen

Folgende Disketten
und Programme bestellt:

_____
_____
_____
_____
_____

am: _____

bei:

Peeker
Softwareabteilung
Postfach 10 28 69
6900 Heidelberg 1

peeker

Software-Karte

Bitte senden Sie mir gegen Rechnung folgende Disketten:

- ☐ Peeker-Sammeldiskette, einzeln
  Disk# _____, Disk# _____
  Disk# _____, Disk# _____
  Preis je Disk DM 28,– (einzeln)
- ☐ Peeker-Sammeldiskette, im Fortsetzungsbezug ab Disk# _____
  (Mindestbezug 6 Disketten)
  Preis je Disk DM 20,–
- ☐ Apple DOS 3.3, Begleitdisk., DM 28,–
- ☐ ProDOS, Band 1, Begleitdisk., DM 28,–
- ☐ ProDOS, Band 2, Begleitdisk., DM 28,–
- ☐ Apple Assembler, Begleitdisk., DM 28,–
- ☐ ProDOS-Editor 1.0, Programm, DM 98,–
- ☐ MMU 2.0, Programm, DM 98,–
- ☐ INPUT 2.0, Programm, DM 98,–
- ☐ DB-Meister, Programm, DM 290,–
- ☐ Superquick, Programm, DM 48,–
- ☐ Turtle Graphics, Programm, DM 98,–
- ☐ Disk 40, Programm, DM 48,–
- ☐ Kyan-Pascal 2.0, Programm, DM 170,–
- ☐ Fast-Writer, DOS 3.3, DM 128,–
- ☐ Fast-Writer, ProDOS, DM 128,–
- ☐ Double-Hires-Tools für Applesoft, DM 28,–
- ☐ Double-Hires-Tools für Kyan, DM 28,–
- ☐ Kyan-Toolkit Nr. _____, DM _____

Datum Unterschrift

Abo-Karte

Name

Firma

Straße

PLZ/Ort

Ich wünsche jährliche Berechnung durch:
☐ Verlagsrechnung ☐ Abbuchung von meinem Bank- bzw. Postscheckkonto

Bank/PschA

Bankleitzahl

Kto.-Nr.

Bitte freimachen

POSTKARTE

Peeker
Leserservice

Dr. Alfred Hüthig Verlag GmbH

Postfach 10 28 69

6900 Heidelberg

Buch-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

Bitte freimachen

POSTKARTE

Peeker
Buchabteilung

Dr. Alfred Hüthig Verlag

Postfach 10 28 69

6900 Heidelberg 1

Software-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

Bitte freimachen

POSTKARTE

Peeker
Softwareabteilung

Dr. Alfred Hüthig Verlag

Postfach 10 28 69

6900 Heidelberg 1

INPUT 2.0

Ein Bildschirm-Maskengenerator für DOS 3.3 und ProDOS

von U. Stiehl

1984, Diskette und Manual, DM 98,–
ISBN 3-7785-1021-5

„Input 2.0" liegt wahlweise in der Bank 1 oder Bank 2 der Language Card und wird durch einen kurzen Driver in den unteren 48K aufgerufen.

Für jedes Feld der Bildschirmmaske lassen sich u. a. definieren: Feldlänge (bis zu 255 Zeichen) – Vtab – Htab – Datentyp (insgesamt 8 Typen) – Scrollflag (starre oder dynamische Maske) – Ctrlflag – Füllflag – Löschflag – Bildschirmflag (40- oder 80-Z-Darstellung). Innerhalb eines Eingabefeldes besteht jeder denkbare Redigierkomfort (Insert, Delete, Rubout, Restore usw.).

Gerätevoraussetzung: Apple IIe oder IIc; ferner Apple II+ im 40-Zeichenmodus

MMU 2.0

Memory Managements Utilities

für die Apple IIe 64K-Karte DOS 3.3 (und ProDOS)

von U. Stiehl

1984, Diskette und Manual, DM 98,–
ISBN 3-7787-1023-1

Insgesamt enthält die neue „MMU 2.0"-Diskette über 25 Programme, die neue Einsatzmöglichkeiten für die Extended 80 Column Card (erweiterte 80-Z-Karte = 64K-Karte für den Apple IIe) erschließen. Ein Teil der Programme laufen auch auf dem Apple II Plus, doch ist „MMU 2.0" primär für 64K-Karte-Besitzer gedacht.

Gerätevoraussetzung: Apple IIe mit 64K-Karte oder IIc

DISK 40

Disketten-Organisationsprogramm für DOS-3.3-35-40 Spuren

von Hermann Seibold und Dipl.-Ing. Udo Marin, 1986, Programmdiskette mit Anleitung, DM 48,–

Durch eine einfach zu bedienende Menüführung können DOS-3.3-Disketten umfangreich bearbeitet oder kopiert werden.

- Tabellarische Ausgabe der Diskettenbelegung
- Ordnen des Catalogs
- „Undelete"n von versehentlich gelöschten Dateien
- Vergleichen von Disketten, Dateien oder DOS-Spuren
- Kopieren von Disketten, Dateien oder DOS-Spuren
- Formatieren von Daten-Disketten
- Erweitern auf 40 Spuren bei bestehenden 35-Spur-Disketten
- Ändern des Boot-Programms
- File-Editor zum Editieren von Disketten-Dateien
- Komfortabler Sektor-Editor für Hex- und ASCII-Darstellung
- VTOC-Editor, z. B. zur Freigabe der DOS-Spuren

Hüthig Software Service, Postfach 10 28 69, D-6900 Heidelberg

Damit sind Sie komplett!

Bestellen Sie jetzt Ihre fehlenden »**peeker**-Hefte«. (Seite einfach kopieren, Heftnummer ankreuzen und Anzahl dazuschreiben)

Preis: DM 6,50 plus Versandspesen. Einfach jetzt bestellen, bevor diese Ausgaben vergriffen sind. Am besten bestellen Sie Ihre praktische »**peeker-Sammelbox**« gleich dazu (Preis DM 16,50 plus Versandspesen).

2/84 ☐

1+2/85 ☐

3/85 ☐

4/85 ☐

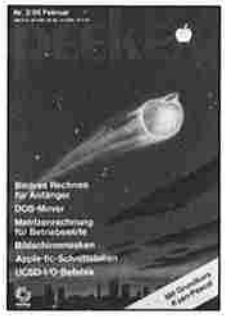
6/85 ☐

7/85 ☐

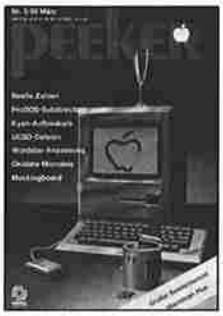
9/85 ☐

12/85 ☐

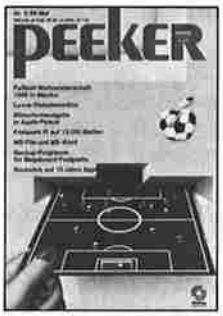
1/86 ☐



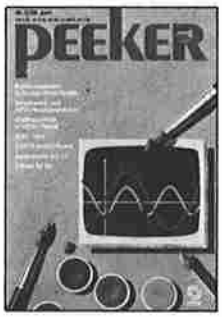
2/86 ☐



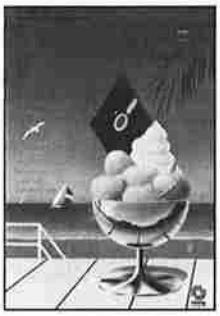
3/86 ☐



5/86 ☐



6/86 ☐



7/86 ☐

8/86 ☐

9/86 ☐

10/86 ☐

11/86 ☐

12/86 ☐

Bitte falzen

In DIN-lang Fensterumschlag einkuvertieren.
Kann als Briefdrucksache verschickt werden.

Hüthig
PUBLIKATION

Antwort

Dr. Alfred Hüthig Verlag
»peeker« Leserservice
Postfach 10 28 69

6900 Heidelberg

Name, Vorname

Straße, Postfach

PLZ, Ort

X

Datum, Unterschrift

Videotext mit dem Apple II

von Dieter Charchot

1. Was bietet Videotext?

Videotext ist ein Informationsdienst von ARD und ZDF, der gemeinsam mit dem normalen Fersehsignal ausgestrahlt wird. Videotext kann jedoch nur mit einem besonderen Decoder benutzt werden, der dieses spezielle Signal verarbeiten kann – oder mit einem Apple IIe und dem Videotext-Baustein VT 1.1.
Wer das Angebot, das Videotext bietet, annimmt, kann zusätzlich zum normalen Fernsehprogramm aktuelle Nachrichten aus allen Teilen der Welt, Informationen zu den verschiedensten Bereichen oder zusätzliche Untertitel zu laufenden Fernsehprogrammen abrufen.

Diese Texte findet man auf „Tafeln", die mit ihren Nummern aufgerufen werden. So enthält beispielsweise die Tafel 100 eine Art Inhaltsverzeichnis der verfügbaren Informationen, aus dem hervorgeht, daß sich Sportnachrichten von Tafel 500, letzte Meldungen von 400, Wetterberichte von 444 usw. abrufen lassen.
Das Angebot ist sehr vielfältig, vom Devisenkurs bis zum Warentest ist alles enthalten; die Informationen sind stets auf dem letzten Stand. Videotext ist kein Ersatz für Zeitschriften oder Fernsehen, sondern versteht sich als Zusatzinformation. Wer einen Vorgeschmack wünscht, kann sich einen Eindruck durch die Nachmittagskostproben von Videotext verschaffen, die auch ohne Zusatzdecoder empfangen werden können, bei denen die Tafeln allerdings vollautomatisch durchlaufen und nicht selbst gewählt werden können.

Im Normalbetrieb ist die jeweils angewählte Tafel nach dem Abruf einer neuen Tafel verloren, es sei denn, man wählt sie wieder neu an. Eine Speichermöglichkeit ist nicht gegeben, Videotext ist ebenso „flüchtig" wie das normale Fernsehbild.

2. Der Apple IIe und VT 1.1

Ganz anders sieht es aus, wenn der Apple IIe mit dem VT 1.1 ausgerüstet wurde. Diese Erweiterung wird vorzugsweise in Slot 2 untergebracht, jedoch kann jeder andere, noch unbesetzte Slot durch einen Einfachsteingriff, der im Manual genau beschrieben wird, in das BASIC-Programm integriert werden.

Die Karte wird mit einem beliebigen Gerät verbunden, das einen Videoausgang besitzt, dann wird die Programmdiskette eingelegt und das System gebootet. Das Programm kann jedoch auch mit „RUN Videotext" gestartet werden. Nach dem Booten wird automatisch die Tafel 100 gesucht und angezeigt. Durch Drücken der Leertaste wird auf eine Übersicht umgeschaltet, die Auskunft über die verfügbaren Befehle gibt, nach einem erneuten Druck der Leertaste gelangt man wieder zurück zur Videotext-Tafel.

2.1. Seitenauswahl

Durch die Eingabe der jeweiligen Tafelnummer im Bereich von 100 bis 899 und anschließendes RETURN wird eine bestimmte Tafel angewählt. Wird die Tafel nicht gefunden, kann die Suche mit „ESC" abgebrochen werden.
Bei dieser Art der Wahl können anstelle von Ziffern auch bis zu drei „*" oder „:" als „Joker" eingegeben werden. Es wird dann die Seite gesucht und gelesen, deren Nummer in den restlichen Ziffern mit der Eingabe übereinstimmt; bei „***" würde die nächste Seite gelesen.
Bei der Eingabe von mehr als drei Ziffern werden die nächsten beiden Ziffern als Code für Mehrfachseiten (Tafeln, bei denen zusammengehörende Informationen auf mehreren Seiten untergebracht sind) interpretiert. Durch „D" (dekrementieren) oder „I" (inkrementieren) kann die nächstgrößere oder -kleinere Seite gewählt werden, „=" liest die gleiche Seite abermals ein. Diese Kommandos müssen jeweils durch RETURN abgeschlossen werden.

2.2. Speichern

Jede eingelesene Tafel wird so lange im Rechner gespeichert, wie der Speicherplatz ausreicht. Der Wert wird jeweils in einer Info-Zeile angegeben. Ist der Speicher voll, wird er beim Einlesen einer weiteren Seite gelöscht und das Spiel beginnt von vorn.
Mit „<" oder „>" kann die jeweils nächste (gespeicherte) Seite „angeblättert" werden, bei der letzten Seite erfolgt ein Überlauf zur ersten und umgekehrt.
Dies bietet sich z.B. nach erledigter „expliziter Vorwahl" an. Mit der E-Funktion können nämlich mehrere Tafeln hintereinander eingelesen werden, ohne daß man die übliche Wartezeit zwischendurch in Kauf nehmen muß. In diesem Modus können bis zu 52 Seiten definiert werden, die dann allerdings sehr kurz sein müssen.

Sinnvollerweise wird bei allen Vorwahlbetriebsarten der Speicher vorher automatisch gelöscht, um möglichst viel Platz zur Verfügung zu stellen.
Dies trifft auch für die zweite Joker-Funktion zu, die durch „J" eingeleitet wird. Danach wird eine Seitennummer mit *einem* „*" eingegeben. Anschließend werden *alle* Seiten eingelesen, die mit dem gewählten Muster übereinstimmen. Bei „1*3" würden z.B. die Tafeln 113, 123, 133 usw. eingelesen.

Der dritte Serienbefehl bezieht sich wieder auf Mehrfachseiten.
Nach „M" wird wieder eine dreistellige Tafelnummer eingegeben. Das Interface filtert automatisch alle zu dieser Tafel gehörenden Folgetafeln mit heraus oder bricht die Suche ab, wenn es sich bei der gesuchten (und gefundenen) Tafel gar nicht um eine Mehrfachseite handelt.
Nervöse Naturen können übrigens mit „G" den Meldeton abschalten, der normalerweise nach jedem Suchvorgang ertönt.

2.3. Verarbeitung der Texte

Sind also glücklich alle Seiten komfortabel gefunden, die im Augenblick von Interesse sind, wird der Hauptunterschied zu einem einfachen Videotext-Decoder deutlich, der bereits in viele TV-Geräte eingebaut ist oder nachgerüstet werden kann.
Zusammen mit dem Apple können die Tafeln auf verschiedene Art und Weise verarbeitet werden. An ersten Stelle steht natürlich die Möglichkeit, die Videotext-Informationen schwarz auf weiß zu dokumentieren. Von Vorteil ist dabei ein möglichst grafikfähiger Drucker.

Da die Videotext-Tafeln jeweils als normale Grafikseite im Speicher stehen, kann zwischen zwei Arten der Druckausgabe gewählt werden: einer normalen HGR-Copy oder der sogenannten SuperCopy, die beim Ausdrucken den Druckerzeichensatz mitbenutzt.
Das vorgefertigte Druckprogramm auf der Systemdiskette ist für den TAXAN KP-810 vorgesehen. Im Manual wird jedoch ausführlich beschrieben, wo welche Änderungen vorgenommen werden müssen, um die Programme (ent-

weder als BASIC-Teil des Hauptprogramms oder als separates Assembler-Druckprogramm) für den eigenen Drucker gefügig zu machen. Ist das geschafft, sind hervorragende Ausdrucke ganz nach Wunsch auf Knopfdruck möglich.

Möchte man jedoch statt dessen von der elektronischen Verarbeitung der Tafeln Gebrauch machen, gibt man entweder „B" (BASIC-Maker), „S" (Speichern) oder „L" (Laden) ein und landet damit im Diskettenmenü, wo man auch noch mit „C" den Diskettencatalog ansehen, mit „D" das Laufwerk wechseln oder mit „L" bestehende Aufzeichnungen löschen kann.

2.4. Der BASIC-Maker

Dabei spielt der BASIC-Maker eine besondere Rolle, denn mit ihm lassen sich Videotext-Tafeln in Textfiles umwandeln. Enthalten die Tafeln also BASIC-Zeilen (jawohl, auch das gibt es!), kann ein Programm in den Computer gelangen, ohne daß man es neu abtippen muß.

Dabei können einige gut durchdachte Kleinigkeiten sinnvoll eingesetzt werden. So kann entschieden werden, ob ein neues Textfile eröffnet werden soll oder ob neue Zeilen an ein bestehendes angehängt werden sollen. Es können Zeilen zu längeren kombiniert oder nach Wunsch auch ganz übersprungen werden, so daß je nach vorliegendem Fall ein Textfile entsteht, das aus BASIC-Zeilen besteht. Apple-Kenner werden schon wissen, daß man das Textfile nur noch mit EXEC starten muß, um es schließlich als Programm in seinem Computer wiederzufinden. Man wird übrigens feststellen, daß Files, die mit dem BASIC-Maker erzeugt wurden, stets automatisch das Kürzel „BM." als Vorsilbe vorangestellt bekommen. Ähnlich verhält es sich mit Videotext-Tafeln. Diese haben auf der Diskette den Vornamen „VT.".

3. Bilanz

Nach all diesen Videotext-Orgien kann mit dem Kommando „C" sowohl der Bildschirm als auch der Speicher gelöscht werden, womit das System wieder frisch initialisiert zu neuen Taten bereit ist. Sollte der frisch gebackene Videotext-Fan allerdings nicht mehr ganz so taufrisch wie sein System sein, wird er sich mit „Q" aus dem Videotext-Programm verabschieden und sich vielleicht überlegen, ob die ganze Angelegenheit wirklich die DM 499,– wert ist, die er für die komplette Ausstattung bezahlt hat.

DB-Meister

Bedienungsanleitung zum Dateiprogramm DB-Meister

Der DB-Meister ist eine Dateiprogramm, das auf allen Apple-II-Typen (+/e/c) läuft. Bei einem Basis 108 oder einem anderen Kompatiblen, bei dem sich Applesoft *nicht* im ROM befindet, laden Sie FPBASIC von der System-Masterdiskette in die lese/schreibfähig gemachte Language-Card (CALL -151, C083 C083, BLOAD FPBASIC, A$D000).
Es gibt *eine* Einschränkung: Das Schemabriefmodul DB-BRIEFER (von Disk #28) funktioniert nur auf einem IIe/c mit 80-Zeichenkarte. Ansonsten läuft das Programm auf allen Geräten mit allen Massenspeichern, bei denen DOS 3.3 (48K-Version, auch Diversi-DOS usw.) verwendet werden kann. Das DOS selbst wird vom DB-Meister *nicht* modifiziert.

Sammeldisk #25 – #27

Die *Hauptmodule* verteilen sich auf die Sammeldisketten #25 bis #27:

Disk #25:
BAUER
BAUER.OBJ
BAUER.EXEC

Disk #26:
STIEHL
DB-MEISTER.OBJ
DB-STARTER.OBJ
DB-PFLEGER.OBJ
DB-BILDSCHIRM

Disk #27:
DB-SORTER.OBJ
DB-SORTER2.OBJ
DB-FILTER.OBJ
DB-DRUCKER.OBJ
DB-PRINTER
LEERKOPF

Vorgehensweise

Diversi-DOS:
Booten Sie DOS 3.3 oder besser Diversi-DOS und formatieren Sie zwei Leerdisketten, die später als Daten- und Programmdisketten benötigt werden.

Datendiskette:
Laden Sie mit LOAD BAUER den Maskengenerator von Disk #25 und entfernen sie zunächst die Zeile 0, die aus Sicherheitsgründen eingefügt worden ist. Falls Sie *keine* 35-Spur-Laufwerke verwenden, so entfernen Sie zusätzlich die Zeilen 540 und 550, die für die Formatierung DOS-loser 35-Spur-Disketten gedacht sind. Speichern Sie anschließend den Maskengenerator mit SAVE BAUER auf die (Kopie der) Sammeldisk #25 zurück.
Nun kopieren Sie mit FID die Dateien BAUER, BAUER.OBJ und BAUER.EXEC auf eine formatierte Leerdiskette. Diese Diskette, die später als Datendiskette fungieren wird, legen Sie nun in Drive 1 und starten Sie den vollautomatischen Standard-Maskengenerator mit EXEC BAUER.EXEC. Nach etwa 1 Minute ist die Datendiskette fertig und enthält die Dateien DB-MASKE, DB-MATCH, DB-DRUCK1, DB-DRUCK2 und DB-DATEI.
(Für *individuelle* Eingabemasken startet man den Maskengenerator direkt mit RUN BAUER, doch sollten Sie erst dann eigene Masken angelegen, wenn Sie mit dem Programm in Verbindung mit der Standard-Eingabemaske vertraut sind.)

Programmdiskette
Auf die andere formatierte Leerdiskette kopieren Sie alle o.g. Dateien von Disk #26 und #27. Falls Sie über zwei 35-Spur-Laufwerke in Slot 6 verfügen, so legen Sie danach die Programmdiskette in Drive 1 und die fertige Datendiskette in Drive 2 und starten Sie den DB-Meister mit RUN STIEHL (Hello-Programm). Es erscheint nun ein Menü, von dem aus Sie das Eingabe-Modul DB-PFLEGER wählen können. Geben Sie hier mindestens 3 Übungsdatensätze ein, da die Sortier- und Druckmodule nur funktionieren, wenn die Gesamtdatei mehr als 2 Datensätze umfaßt. Näheres zu den Modulen DB-PFLEGER, DB-SORTER, DB-FILTER und DB-DRUKKER können Sie der Anleitung entnehmen.

Andere Datenträger
Wenn Sie über eine Festplatte, RAM-Karte oder größere Laufwerke verfügen, so müssen Sie in dem Hello-Programm STIEHL eintragen, wo das DB-Programm
a) die Programm-Module (DB-MEISTER.OBJ usw.) und
b) die Datendateien (DB-MASKE usw.)
suchen soll (Slot, Drive, Volume)
Programm- und Datendateien können sich auch auf einer einzigen Diskette befinden.

Sammeldisk #28

Die Disk #28 kann *nur* von Fortsetzungsbeziehern bestellt werden und wird im übrigen *nicht* automatisch verschickt. Sie enthält
a) Quelltexte zu den 6502-Maschinenprogrammen der Hauptmodule
b) Quelltexte zu den TASC-compilierten Applesoft-Programmen
c) DB-Hilfsprogramme (Applesoft- und Maschinenprogramme)
d) Quelltexte zu den 6502-Maschinenprogrammen der Hilfsmodule
e) Hilfsmenüs zum Maskengenerator
f) RAM-Disk-Driver für IIe/c-Besitzer mit 128K
g) Musterbeispiel für ein mögliches Rechen-Modul (muß individuell an die jeweilige Maske angepaßt werden!)
Zusätzlich zur Sammeldisk #28 werden das Originalmanual sowie die Originalergänzungsanleitung geliefert. Preis für Disk #28 plus beide Manuals nur DM 20,- für Fortsetzungsbezieher. Die Bestellung muß spätestens am 20.3.1987 beim Hüthig Software Service eingehen. Da nur noch ca. 100 Manuals vorrätig sind, werden die Bestellungen in der Reihenfolge des Eingangsdatums bearbeitet.
Die Disk #28 brauchen Sie nur, wenn Sie an den diversen Hilfsmodulen und/oder an den Quelltexten der Programme und/oder an einer vollständigen Anleitung interessiert sind. Für einfachere Anwendungen genügen die Hauptmodule von Disk #25 – #27 in Verbindung mit der nachfolgenden Teilanleitung.

Disk #28:
A-
T.DB-HELFER.OBJ
DB-HELFER.OBJ
T.DB-BILDSCHIRM
DB-BILDSCHIRM
T.DB-PRINTER
DB-PRINTER
B-
DB-PFLEGER
DB-SORTER
DB-SORTER2
DB-FILTER
DB-DRUCKER
C-
DB-HELFER
DB-MATCH-AENDERER
DB-MATCH-AENDERER.OBJ
DB-LESER
DB-LESER.OBJ
DB-LOESCHER
DB-STATUS
DB-TAUSEND
DB-DRUCKABBRUCH
DB-NUMMERNSORT
DB-BRIEFER.OBJ (nur e/c!)
DB-BRIEFER1.OBJ
DB-BRIEFER2.OBJ
DB-BRIEF
D-
T.DB-MATCH-AENDERER.OBJ
T.DB-LESER.OBJ
E-
DB-BAUER.ANLEITUNG
DB-BAUER.OBJ
F-
RAMDISK-S4D1
G-
DB-RECHNER.BEISPIEL

1. DB-Pfleger

1.1. Menü-Optionen

a) Erst-Menü-Optionen:

1 Neueingabe
3 Löschen
5 Einzel-Ändern
7 Gesamt-Ändern
9 Drive-Wechsel
0 Ende und Zweit-Menü

b) Zweit-Menü-Optionen:

2 DB-Starter
4 Datei-Wechsel
6 Zurück zum Erst-Menü
8 Programm-Ende

1.1.1. Neueingabe (1)

Bei der Neueingabe wird die Recordnummer automatisch vergeben. Wenn die Diskette voll ist, müssen Sie eine neue, bereits vorformatierte Datendiskette mit derselben Maske einlegen und Datei-Wechsel wählen. An der jeweils zuletzt vergebenen Record-Nummer kann man ersehen, wie voll die Diskette bereits ist (es sei denn, daß zwischendurch gelöscht wurde. In diesem Fall werden zunächst die „Löcher" mit neuen Records aufgefüllt).

Nach Eingabe in den eigentlichen Feldern kann man noch beim freien Match 1-4 beliebige Zeichen als Selektionsmerkmale eingeben. Match 1 und Match 2 werden automatisch ausgewiesen.

1.1.2. Löschen (3)

Beachten Sie, daß das Löschen von Records nicht mehr rückgängig gemacht werden kann, da der einmal gelöschte Record durch den nächsten, neuen Record überschrieben wird. Ein zu löschender Record wird wie beim Einzeländern gesucht. (Aus Sicherheitsgründen gibt es kein Gesamt-Löschen.)

1.1.3. Einzel-Ändern (5)

Das Suchen eines Records (zwecks Änderung oder Löschung) kann geschehen nach:

1. Recordnummer
2. Matchcodes

Wenn man die Frage NUMMER J/N mit J beantwortet, wird nach Recordnummer, ansonsten nach Matchcodes gesucht.

Wenn man nach der Recordnummer sucht, gibt man diese Recordnummer in demselben Feld ein, wo man sonst das freie Match eingibt.

Man beachte, daß gegenüber anderen, ähnlichen Programmen die Suche in beiden Fällen praktisch gleich schnell vonstatten geht. Im ungünstigsten Fall dauert die Suche 1/2 Sekunde.

Bei einem Matchcode wie z.B.

a2bm Stie 6900

können Sie suchen nach

```
a2bm Stie 6900   Match 0 Match 1 Match 2
a2bm
     Stie
          6900
a         6
      ti          usw.
```

Leertasten fungieren also als Wildcard-Characters. Beachten Sie jedoch, daß die Art und die Position der Match-Zeichen kritisch ist. Z.B. wird der Record mit „Stie", „Sti ", „St ", „S ", „ tie", „ ti " und „ i " gefunden, nicht jedoch mit „stie" (Kleinbuchstabe) oder mit „ st" (falsche Position).

1.1.4. Gesamt-Ändern (7)

Wenn Ihnen ein Matchcode nicht genau bekannt ist, dann wählen Sie Gesamtändern, da beim Einzeländern nach dem ersten erfolgreichen Suchen nicht mehr weitergesucht wird. Mit Gesamt-Ändern kann man eine ganze Datei nach einem bestimmten Match-Merkmal durchsuchen. Wenn ein gefundener Record am Bildschirm erscheint, wird man gefragt ÄNDERN J/N. N bewirkt Weitersuchen. Hat man jedoch J getippt, dann wird man nach der Änderung WEITER J/N gefragt. N beendet die Gesamtsuche.

Wenn man bei Gesamt-Ändern eine Record-Nummer (nach J bei NUMMER J/N) eingibt, werden alle Records ab dieser Nummer am Bildschirm zwecks Änderung angezeigt.

1.1.5. Ansehen

Das reine Ansehen eines Records ist keine Menü-Option. Wählen Sie zum Ansehen Gesamt-Änderung und geben Sie die Matchcodes so genau wie bekannt ein. Wenn der Record dann am Bildschirm erscheint, tippen sie N für ÄNDERN J/N, oder wählen Sie Einzel-Ändern und tun Sie so, als wollten Sie ändern. Ab dem ersten Feld lassen Sie den Finger auf der RETURN-Taste, womit der Cursor nach unten saust, ohne irgend etwas zu ändern.

1.1.6. Drive-Wechsel (9)

Vom Erst-Menü aus können Sie mit 9 Drive-Wechsel wählen. Um Datei-Wechsel zu wählen, tippen Sie erst 0 (für Ende und Zweit-Menü) und dann 4 für Datei-Wechsel. Wenn Sie entweder nur 1 Drive haben oder zu Beginn des DB-Pflegers bei Anzahl der Datendisketten 1/2 1 angegeben haben, passiert bei 9 = Drive-Wechsel gar nichts.

a) Datei-Wechsel bedeutet, daß die Matchcodes UND die DB-Maske einer ANDEREN Datei eingelesen werden.

b) Drive-Wechsel (= Wechsel des Arbeitsdrives) bedeutet, daß Sie von einem zum anderen Laufwerk umschalten, wobei sich in beiden Drives Datendisketten mit DENSELBEN Bildschirmmasken befinden müssen. Es werden NUR die Matchcodes der zweiten Datendiskette eingelesen.

2 Drives und Dateien mit mehr als 2 Disketten: Legen Sie die z.B. dritte Diskette in dasjenige Drive, das gerade nicht Arbeitsdrive ist und wählen Sie Drive-Wechsel.

1 Drive und Dateien mit mehr als 1 Diskette: Legen Sie die z.B. zweite Diskette in Drive 1 und wählen Sie Datei-Wechsel.

Der Drive-Wechsel (ca. 8 Sekunden) ist etwas schneller als der Datei-Wechsel (ca. 13 Sekunden). Beim Drive-Wechsel wird aus Zeitgründen nur PARTIELL geprüft, ob die andere Diskette DASSELBE Dateiformat hat. SIE müssen durch Etikettierung der Datendisketten sicherstellen, daß Sie niemals aus Versehen eine Datendiskette mit einem ANDEREN Maskenformat einlegen, sonst kann bei verschiedenem Maskenaufbau diese falsche Datendiskette zerstört werden!

Faustregel:

1. Im Zweifelsfall Datei-Wechsel wählen. Dies ist absolut narrensicher.
2. Datendisketten mit derselben Dateistruktur in einer eigenen Diskettenbox aufbewahren und niemals mit anderen Disketten mischen.

1.1.7. Ende (0)

Verlassen Sie den DB-Pfleger grundsätzlich nur über Ende. Wenn Sie 0 getippt haben, kommen Sie in das Zweit-Menü. Hier können Sie entweder über 8 Programm-Ende endgültig das Programm verlassen oder über 2 DB-Starter zu einem anderen Programmteil (DB-Sorter usw.) wechseln.

1.2. Totalsuche

Das Kommando für Totalsuche lautet Ctrl-T. Dieser Befehl steht nicht im Erst-Menü, damit die Totalsuche nur dann angewandt wird, wenn von einem Record weder die Matchcodes noch die Recordnummer bekannt sind.
Nach Ctrl-T kann man wählen zwischen Gesamtändern und Einzeländern. Danach gebe man in den Recordfeldern - also hier nicht in den Matchfeldern - diejenigen Zeichenfolgen ein, an die man sich noch erinnert. Wenn man sich z.B. noch an die Telefonnummer oder den Straßennamen erinnert, gebe man sie in den entsprechenden Feldern ein. Wenn im Falle einer Adreßverwaltung die Anschrift z.B. lautet:

Ulrich Stiehl
Lörracherstraße 3
6900 Heidelberg 1 23.12.1947

und als Matchcodes nur Stie(hl) und 6900 verschlüsselt, aber im Moment unbekannt sind, dann kann man bei der Totalsuche z.B. nach Lörracherstraße oder nach 23.12.1947 suchen.

Die Totalsuche findet JEDE Zeichenkette, wenn sie an der richtigen Feldposition eingegeben wird, doch dauert die Suche u.U. 1 Minute und mehr, weil die gesamte DB-Datei auf der Diskette durchsucht werden muß.

2. DB-Sorter

Der DB-Sorter dient zum Sortieren des Inhalts EINER Diskette. Wenn also eine Datei mehr als 1 Diskette umfaßt, muß sie VON ANFANG AN sinnvoll aufgeteilt werden, z.B. nach Inland/Ausland. Das Sortieren von zahlreiche Disketten umfassenden Großdateien ist bei 2 Drives nicht zu automatisieren und wurde deshalb nicht implementiert.

Die Sortierung dauert - bei z.B. 600 Records - mit Einlesen und Speichern ca. 1 Minuten. Bei der Sortierung wird auf der Programmdiskette die Datei SORTZAHLEN aktualisiert, die für den DB-Drucker erforderlich ist. Auf der Programmdiskette steht außerdem die Datei SORTSTRINGS, die bei der Sortierung als Zwischenspeicher dient.
Diese beiden Dateien dürfen niemals gelöscht werden.

2.1. Totalsort

Der DB-Sorter bietet als Standardoption TOTALSORT J/N an, da dies die üblichste Sortierform ist. Wenn man diese Frage mit J beantwortet, erfolgt die Sortierung vollautomatisch. Totalsort bedeutet, daß ALLE Records in der Reihenfolge Match 1, Match 2, Match 0 sortiert werden.

2.2. Selektionssort

Wenn man Totalsort J/N mit N beantwortet, muß man zunächst die Untersortierreihenfolge festlegen und kann im Anschluß daran durch Eingabe in den umgeordneten Matchfeldern eine Selektierung vornehmen.

Das Sortieren geschieht nach den drei Matchcodes Match 0 (= freies Match), Match 1 und Match 2, wobei eine Untersortierung in beliebiger Reihenfolge möglich ist (6 Kombinationen). Läßt man die Matchfelder selbst leer, werden ALLE Records sortiert, ansonsten nur diejenigen Records, auf die die Matchcodes zutreffen. Auf diese Weise kann man für den späteren Ausdruck gezielt Records auswählen, z.B. alle Adressen mit der Postleitzahl 6900 oder alle Applebesitzer usw.

Beispiel:

Match 0 enthalte als erstes Zeichen bei den zutreffenden Adressen ein „g" für guter Kunde
Match 1 enthalte jeweils die ersten 4 Stellen des Firmennamens
Match 2 enthalte jeweils die 4 Stellen des PLZ-Feldes

Also z.B. Firma Meier in Berlin als guter Kunde:

Match 0	Match 1	Match 2	
g	Meie	1000	
0.	1.	2.	Stelle
nullte	erste	zweite	Stelle

Es soll nun sortiert werden nach PLZ und innerhalb PLZ nach Firmenname und an letzter Stelle nach g.
Match 0 soll also an die letzte = 2. Stelle rücken, d.h. wir geben bei Match 0 eine 2 ein.
Match 1 soll an mittlerer = 1. Stelle bleiben, wir geben also bei Match 1 eine 1 ein.
Match 2 soll an die nullte = faktisch erste Stelle rücken, d.h. wir geben bei Match 2 eine 0 ein.
Nachdem wir die Ziffern 2, 1, 0 eingeben haben, sehen wir am Bildschirm die Vertauschung:

Match 2: Match 1: Match 0: g

Nun geben wir noch in dem jetzt ganz rechts stehenden Match 0 ein g, so daß nur die g(uten) Kunden sortiert werden. Danach kann die Sortierung beginnen.

2.3. Sortiernormierung

Beim DB-Pfleger werden bei Match 1 und 2 die Anfangszeichen wie folgt normiert:

Kleinbuchstaben	-»	Großbuchstaben
Umlaute	-»	Nichtumlaute
Sonstige Zeichen	-»	?
ß wird ebenfalls zu		?

Matchcodes, die Sonderzeichen oder Ziffern enthalten sowie reine Zahlen mit ungleicher Stellenzahl werden nicht richtig sortiert.
Üblicherweise stellt man deshalb das Match 0 bei der Sortierung an die 2., d.h. letzte Stelle, da die Anfangszeichen des freien Match 0 nicht modifiziert werden.

Hinweis: Beim DB-Pfleger können Sie nach Karteileichen (= unzureichend ausgefüllten Adressen) suchen, indem Sie ? bei den Matchcodes 1 und 2 eingeben.

3. DB-Filter

Der DB-Filter gestattet ZUSÄTZLICH zum Sortieren das KUMULIERTE = mehrfach wiederholte Herausfiltern (a) einzelner Match-Merkmale sowie (b) beliebiger Felder-Merkmale.
Der DB-Filter hat also ALLE Funktionen des DB-Sorters und darüber hinaus die erwähnten Filtermöglichkeiten. Der DB-Filter wurde als ein eigenes Programm-Modul angelegt, weil er etwas komplizierter als der DB-Sorter ist.

3.1. Kumuliertes Filtern von Matchfeldern (= Matchfiltern)

Hätte man z.B. im freien Match 0 bei einer Kundendatei durch die Buchstaben A, B, C usw. Kundenmerkmale verschlüsselt, dann könnte man mit dem DB-Sorter nur entweder A oder B oder C usw. herausfiltern oder müßte eine Totalsortierung vornehmen. Mit dem DB-Filter lassen sich jedoch z.B. A und B ohne C oder B und C ohne A herausfiltern und anschließend sortieren. Damit sind Selektionen aller Spielarten realisierbar.
Mit dem DB-Filter läßt sich sogar ein einzelner Record gezielt selektieren, wenn man die entsprechenden Matchcodes eingibt. Dies empfiehlt sich z.B. dann, wenn bestimmte Records erneut ausgedruckt werden sollen.
Das einzelne Matchfiltern dauert nur ca. 0,2 Sekunden.

3.2. Kumuliertes Filtern von Recordfeldern (= Totalfiltern)

Das Totalfiltern entspricht der Totalsuche beim DB-Pfleger. Wenn man TOTALFILTERN J/N wählt, dann kann man in den Recordfeldern der Bildschirmmaske einen beliebigen Suchstring eingeben. Nehmen wir an, daß Postleitzahl und Zuname Matchfelder seien, man jedoch (zusätzlich) nach Straße filtern möchte. Würde man dann z.B. im Straßenfeld als erstes Zeichen S eingeben, dann würden alle Records herausgefiltert, bei denen der Straßenname mit einem großen S beginnt, z.B. Stresemannstraße, Siedlungsstraße usw. Ein Filtern nach Straßenname wäre natürlich normalerweise nicht besonders sinnvoll. Das Totalfiltern wird deshalb üblicherweise dann angewandt, wenn aufgrund diffiziler Selektionskriterien das freie, 4stellige Match 0 zum Selektieren nicht ausreicht. In diesem Fall wird INNERHALB der Bildschirmmaske ein zusätzliches, 1 - 40 Zeichen langes Selektionsfeld eingerichtet, das hinsichtlich des Totalfilterns dieselbe Funktion übernimmt wie das freie Match 0 hinsichtlich des Matchfilterns.
Das Totalfiltern dauert wie die Totalsuche ca. 1 Minute und mehr.

4. DB-Drucker

Vor dem Ausdruck muß immer erst sortiert werden, es sei denn, daß seit dem letzten Ausdruck keine Records geändert worden sind. Der DB-Drucker liest zunächst die SORTZAHLEN ein und fragt dann im ersten Menü nach den drei möglichen Ausdruckformen:

1 Etiketten (mit Feldzusatz)
5 Tabellen (mit Tabellenkopf)
9 Schemabriefe (Adressen + Brief)

Der DB-Drucker ist relativ kompliziert, weil er alle möglichen Sonderwünsche hinsichtlich der Ausdrucksformen berücksichtigt. Deshalb soll er hier Schritt für Schritt behandelt werden.

4.1. Druckmaske

Druckmasken werden bei Etiketten, Tabellen und Briefen benötigt. Es können 2 verschiedene Ausdruckmuster namens DB-Druck1 und DB-Druck2 auf der Datendiskette abgespeichert werden. Das Abspeichern der Dateien DB-Druck1 und DB-Druck2 geschieht durch den DB-Drucker automatisch.

Zur Anlage der Ausdruckschablonen müssen Sie die gewünschten Parameter eingeben. Sie sehen am Bildschirm eine Schablonentabelle mit den Inhalten:
N = Nummer
F = Feld
L = Länge des Felds
R = Reihenfolge
V = Vorschlag
A = Abstand links

4.1.1. Reihenfolge der Felder

Wenn Sie in der Zeile mit der Nummer 1 eine 5 eingeben, dann bedeutet dies, daß als 1. Feld das Feld 5 ausgedruckt werden soll. Wenn Sie in der Zeile mit der Nummer 2 eine 1 eingeben, dann bedeutet dies, daß als 2. Feld das Feld 1 ausgedruckt werden soll usw.
Bei Feld 0 = Recordnummer geben Sie 99 oder 2 Leertasten ein, wenn die Recordnummer nicht ausgedruckt werden soll. Sie geben hier eine 0 ein, wenn die Recordnummer ausgedruckt werden soll. (Die Recordnummer kann nur als physisch erstes Feld ausgedruckt werden.)
Es muß mindestens 1 Feld (außer Feld 0) ausgewählt werden. Die Eingabe wird mit 99 oder zwei Leertasten abgebrochen. (Dies gilt nicht für Feld 0.)

4.1.2. Vorschlag (RETURN)

Der Vorschlag bedeutet Return oder Zeilenvorschub VOR dem auszudruckenden Feld. Beispiel:
Herrn
Fritz Meier
Fritz hat den Vorschlag 1, weil nach Herrn und vor Fritz 1 Return erfolgte. Meier hat dagegen den Vorschlag 0, steht mithin auf derselben Zeile wie Fritz. Vorschlag 0 bedeutet also selbe Zeile, Vorschlag 1 nächste Zeile, Vorschlag 2 übernächste Zeile, also 1 Leerzeile usw.

Sollen die Records nahtlos, also ohne Zwischenraum gedruckt werden, dann muß das erste physische Feld jedes Records den Vorschlag 0 haben. Soll dagegen zwischen den Records jeweils 1 Leerzeile sein, dann muß das erste physische Feld jedes Records den Vorschlag 1 haben, usw. Bei Etiketten benötigt man üblicherweise für das erste physische Feld einen Vorschlag von etwa 6, also 6 Leerzeilen.

4.1.3. Abstand links

Abstand bedeutet Anzahl der Leertasten zur linken Blattkante oder, wenn mehrere Felder in derselben Zeile stehen, Abstand zum vorangehenden Feld. Beispiel:
——Dr.—Fritz—Müller
Dr. hat den Abstand 2 (hier durch 2 Striche gekennzeichnet), Fritz und Müller haben jeweils den Abstand 1.

Hinweis: Bei Etiketten werden leere Felder in derselben Zeile nicht gedruckt. Stattdessen werden die nachfolgenden Felder nach links gerückt, indem von dem vorangehenden Abstand der nachfolgende Abstand abgezogen wird. Nehmen wir an, Fritz Müller habe keinen Dr., dann gilt —— minus — = —, mithin wird anstelle von ——Dr. nur — gedruckt. Daraus folgt, daß Abstand des nachfolgenden Feldes kleiner oder gleich Abstand des vorangehenden Feldes sein muß, weil der Abstand nicht negativ werden darf. Angenommen wir hätten Dr.-Fritz-Müller, also vor Dr. den Abstand 0, dann würde, wenn Fritz Müller keinen Dr. hätte, —Fritz-Müller gedruckt, Fritz würde also nicht an der linken Blattkante beginnen.

Zusammenfassendes Beispiel:

Nr.	Feld	Länge	Reihenfolge	Vorschlag	Abstand links
0	R:	4	99	99	99
1	VOR	20	1	5	2
2	ZUN	20	2	0	1
3	STR	25	6	1	2
4	PLZ	4	4	1	2
5	ORT	25	5	0	1
6	TEL	15	99	99	99

Dies ergäbe beim Etikettendruck dann durch jeweils 5 Leerzeile getrennte Records in der Form:

	ausgedruckte Felder	
Ulrich Stiehl	1	2
06221/301141	6	
6900 Heidelberg 1	4	5

4.2. Brief und Tabellenkopf

Schemabriefe und Tabellenköpfe können mit einem beliebigen Textverarbeitungsprogramm geschrieben werden, das normale DOS 3.3 Textfiles erzeugt. Dies ist bei den meisten Textverarbeitungsprogrammen der Fall, z.B. Applewriter IIe usw. CP/M-Dateien, speziell von Wordstar, können mit dem CPMXFER Programm des CP/M A.L.D.S. Programm-Pakets in DOS 3.3 Textfiles umgewandelt werden.

Briefe und Tabellenköpfe können eine maximale Länge von 4095 Zeichen haben. JEDE Zeile des Briefes oder des Tabellenkopfes muß mit einem RETURN abgeschlossen werden. (Endlos-Schreiben ist also nicht zulässig.) Durch die Returns hat man im übrigen eine bessere Kontrolle über das Aussehen des Schemabriefes mit den möglichen Feldeinschüben.
Briefe und Tabellenköpfe können Kontroll-Zeichen zur Druckersteuerung für fette Schrift usw. enthalten.

Wenn man beim Erst-Menü des DB-Druckers Briefe (5) oder Tabellen (9) wählt, muß sich bereits ein unter einem beliebigem Namen abgespeicherter Brief oder Tabellenkopf auf der DB-Programmdiskette befinden.

4.3. DB-Printer

Der DB-Printer ist ein Binärfile auf der Programmdiskette, der ein kundenspezifisches Driver-Programm für den Drucker enthalten kann. Der DB-Printer enthält z.Zt. ein kurzes Assemblerprogramm, das ESC-Kontroll-O an den Drucker schickt, um den Seitenendeschalter abzustellen.

4.4. Etiketten

Der Ausdruck muß auf einbahnigen Endlosetiketten erfolgen, wobei beim Drucker der SEITENENDESCHALTER abgestellt sein muß (siehe oben).

Neben Adreßetiketten kann man auch formlose, nicht-tabellarische Listen auf Endlospapier ausdrucken.

Beim Etikettendruck werden Leerräume am Feldende (nicht jedoch am Feldanfang) automatisch elimiert und durch den vorgegebenen Abstand ersetzt:

Ulrich Stiehl		etikettenmäßig
Ulrich	Stiehl	tabellenmäßig

4.4.1. Feldzusatz

Beim Etikettendruck - wie im übrigen auch beim Briefedruck - ist 1 automatischer Feldzusatz möglich, der dann für alle Records gilt, z.B. Geschäftsleitung o.ä. Der Feldzusatz kann nicht vor der Recordnummer stehen. Ferner kann er selbstverständlich nicht vor einem Feld stehen, das aufgrund der definierten Druckmaske gar nicht ausgedruckt wird.

4.4.2. Justierung

Das sich Etikettenbahnen erfahrungsgemäß nicht einfach justieren lassen, wird bei der ersten ausgedruckten Adresse solange »1 = weiter 0 = wiederholen« angezeigt, bis man 1 getippt hat. Man gehe dabei so vor: Wenn die erste Adresse nicht richtig auf dem Etikett plaziert ist, stelle man den Drucker auf OFF-LINE und korrigiere manuell mit der LINE-FEED-TASTE. Dann stelle man den Drucker wieder auf OFF-LINE und tippe 0 für Adreßwiederholung. Wenn die Adresse dann richtig plaziert ist, tippe man 1 für weiter.

4.5. Tabellen

Bei tabellarischem Druck muß zuvor die entsprechende Kopfdatei eingeladen werden. Die Kopfdatei darf kein »&« enthalten, da dies als Steuerzeichen für Briefe verwendet wird.

Bei Tabellendruck muß man spezifieren, nach wieviel Records (also nicht nach wieviel Zeilen) ein Formfeed (= Blattvorschub) erfolgen soll. Ferner kann man noch spezifizieren, ob und wie lang eine mögliche Trennlinie zwischen den Records sein soll. Gibt man 0 bei Trennlinienlänge ein, wird keine Trennlinie gedruckt, ansonsten eine Trennlinie von 1-99 Strichen. Der Tabellendruck ist nur für Endlospapier vorgesehen.

4.6. Schemabriefe

Ein Schemabrief darf 4095 Zeichen lang sein (17 Sektoren) und kann mit dem Applewriter II oder jedem anderen Textverarbeitungsprogramm, das Textfiles erzeugt und normale ASCII-Codes benutzt, erstellt werden. Es sind folgende Regeln zu beachten:

1. JEDE Zeile muß mit einem Return abschließen. (Zusätzliche Returns für Absätze usw. sind erlaubt.)
2. Als linker Rand gilt der Abstand links, den man bei der Anrede eingegeben hat.
3. Zweiseitige Briefe müssen nach der ersten Seite ein Ctrl-L für Formfeed enthalten. Am Briefende erfolgt das Ctrl-L automatisch.
3. Der Brief darf keine Anrede, auch keine SCHEMAANREDE: »Sehr geehrte Herrn« o.ä., enthalten.

4.6.1 &-Einschub

Das & innerhalb des Schemabriefes dient als Steuerzeichen und darf deshalb nicht als druckbares Zeichen vorkommen. (Es darf jedoch in einem Record selbst, z.B. Meier & Söhne, vorkommen.)

Ein Schemabrief kann aus 3 konstanten Textteilen bestehen:

1. Briefkopf
   (danach &-Anschrift)
2. Betreff
   (danach &-Anrede)
3. Briefkörper (eigentlicher Text)
   (darin ggf. &-Einschübe)

4.6.1.1 Erstes & (= Anschrift)

Soll der Brief einen Kopf erhalten, so schreibe man den Kopf und schließe daran das erste & an. Vor und nach dem ersten & können Returns stehen. Das erste & steht für die Einschubstelle der Anschrift, die hier automatisch eingefügt wird. Hat der Brief keinen Kopf, so beginnen man den Brief mit einem nackten &.

4.6.1.2 Zweites & (= Anrede)

Hat der Brief einen Betreff, ein Datum oder ähnliches, das zwischen Adresse und Anrede stehen kann, dann schreibe man nach dem Betreff das zweite &.
VOR dem zweiten & können Returns sein. NACH dem zweiten & muß jedoch unmittelbar dasjenige Satzzeichen stehen, das nach der Anrede gedruckt werden soll, da das zweite & die Anrede-Einschubstelle markiert.
Hat der Brief weder Kopf noch Betreff, so beginne man den Brief stets mit &&! oder &&, - je nachdem, welches Satzzeichen nach der Anrede erwünscht ist.

4.6.1.3 Drittes und weiteres &

a) & allein (= Tastatureinschub)

Das dritte und jedes weitere & im Brief gestattet einen jeweils bis zu 39 Zeichen langen individuellen Einschub, der über die Tastatur erfolgen muß. Leertasten am ENDE des Einschubs werden ignoriert. Die Zeile darf mit Einschub die drucktechnisch mögliche Zeilenbreite nicht überschreiten.

Zwei oder mehrere unmittelbar aufeinanderfolgende & (z.B. &&&) sind erlaubt, falls der Einschub länger als 39 Zeichen sein soll.

Beispiel:

...und senden Ihnen beigefügt je ein Muster der neuen Stoffe
& und &.
Wir hoffen, daß...

Das Drucker-Interface muß auf Bildschirm-Echo eingestellt sein, damit man den Brief sieht, in den der Einschub erfolgen soll.

b) & + Feldnummer (= Feldeinschub)

Wenn unmittelbar nach & eine zweistellige Ziffer steht (&00, &01...&25), dann wird an dieser Stelle im Brief das Feld des jeweiligen Records eingefügt.

Beispiel:
...wie Sie ja, sehr geehrter Herr &01, wissen...

(&01 bezieht sich auf das Feld 1; hier Namensfeld)

Beachte: Zwischen & und Feldnummer darf kein Zwischenraum sein. Feldnummern « 10 sind in der Form &00 ... &09 zu schreiben. Wenn das betreffende Feld leer ist, erfolgt kein Ausdruck, d.h. auch kein Leerraum.
Abschließend sei darauf hingeweisen, daß JEDER Brief mindestens 2 & enthalten muß, im Grenzfall in der Form &&!

4.6.1.4. Rechnungen und Lieferscheine

Der & + Feldnummer-Einschub eignet sich neben Briefen auch für Rechnungen, Lieferscheine, Mahnungen usw.
Da der DB-Meister selbst nicht rechnet, kann ein DB-Rechner als zusätzliches Programm-Modul geliefert werden, mit dessen Hilfe die Berechnungen für die Fakturierung usw. sowie für die Zahlenformatierung vorgenommen werden können. Der DB-Meister ist dank der Datentyp-Definition darauf bereits vorbereitet.

4.6.2. Individuelle und Schemaanrede

Eine individuelle Anrede setzt sich aus 3 Teilen zusammen:
(1) Anrede(teil), z.B. Sehr geehrter Herr,
(2) recordspezifisches Namensfeld (z.B. Müller), und
(3) Komma (muß im Applewriter-Brief enthalten sein),
z.B.: Sehr geehrter Herr Müller. Der Anredeteil kann 30 Zeichen lang sein.

Eine Schemaanrede, die ebenfalls 30 Zeichen umfassen kann, z.B. Sehr geehrte Herren, wird verwandt, wenn die angeredete Person nicht namentlich bekannt ist.

Der DB-Drucker verlangt stets 2 Anreden:

1. Anrede = gemeint ist Anredeteil
2. Ersatz = gemeint ist Schemaanrede

Die Ersatzanrede wird stets dann verwandt, wenn man bei der Feldnummer 0 eingibt oder wenn das Namensfeld leer ist. Wenn eine der zwei Anreden nicht vorkommt, dann geben Sie XXX o.ä. ein.

Im einzelnen müssen zum Anredekomplex folgende Angaben gemacht werden:

a) Anrede N:

= Nummer des Feldes, vor dem die Anrede stehen soll.

b) Anrede-Vorschlag V:

= Anzahl der Returns (1-15) zwischen Adreßfeld und Anrede (oder zwischen Adreßfeld und Betreff)

c) Linker Rand oder Abstand A:

Achten Sie darauf, daß Adresse, Anredeteil, Brief und &-Einschub stets denselben linken Rand aufweisen.

d) Anrede»Leertaste«Feld J/N

Diese Frage beantworte man mit J, falls zwischen Anrede und Namensfeld 1 Leertaste stehen soll (z.B. Lieber Otto)

Bei spezifischen Anredefeldern in der Form:

Sehr geehrte + r Herr Müller
Sehr geehrte + —Frau Mayer

darf zwischen Anrede und Feld keine Leertaste sein. (Der Strich bei —Frau steht hier für Leertaste.)
Aus diesem Beispiel wird übrigens ersichtlich, daß beim nicht-tabellarischen Druck Leertasten ZU BEGINN eines Feldes niemals, AM ENDE eines Feldes dagegen stets eliminiert werden.

4.6.3. Struktur eines Schemabriefes

a)	mindestens 1 Return	
b)	Briefkopfzeile(n)	
c)	mindestens 1 Return	
d)	& = Adresseneinschub	obligatorisch
e)	mindestens 1 Return	
f)	Betreffzeile(n)	
g)	mindestens 1 Return	
h)	&, = Anredeeinschub	obligatorisch
i)	mindestens 1 Return	obligatorisch
j)	Briefkörper 1. Seite	obligatorisch
	1. Zeile + 1 Return	
	2. Zeile + 1 Return	
	usw.	
k)	Ctrl-L	
l)	Briefkörper 2. Seite	
	1. Zeile + 1 Return	
	2. Zeile + 1 Return	
	usw.	

Wenn kein Briefkopf vorkommt, entfallen die Position a - c.
Wenn kein Betreff vorkommt, entfallen die Positionen e - g
Wenn der Brief nur 1. Seite umfaßt, entfallen die Positionen k - l.
Es bleiben dann noch die Positionen d, h, i und j.

Wenn Sie als als Drucker-Slot 0 wählen, erfolgt der Ausdruck am Bildschirm. Dies empfiehlt sich als Test, wenn man ein neues Ausdruckformat ausprobieren will.

4.6.4. Einzelblatt

Bei Briefen besteht die Option Einzelblatt J/N. Hat man J getippt, so hält der DB-Drucker nach dem Formfeed am Ende eines jeden Briefes an. Wenn man jetzt 1 tippt, wird der nächste Brief gedruckt, wenn man 0 tippt, wird derselbe Brief wiederholt.

Ausgabe und Eingabe mit TYPETERM®

im Slot Ihres

APPLE II/IIe/IIgs

Das bedeutet: Computertextverarbeitung von der Schreibmaschinentastatur! Steckerfertig ohne Umbau.

Die neue CE-550!
mit TYPETERM DM 1.318,–

TYPETERM-Interface DM 399,–
für alle BROTHER-Typenradschreibmaschinen ab AX-30 bis EM-811
(auch für Vorgängermodelle!)
Paketpreis z. B.:

EM-501 mit TYPETERM ..... DM 2056,–
EM-511 mit TYPETERM ..... DM 2332,–
EM-701 mit TYPETERM ..... DM 2388,–

TYPETERM – ein starkes Interface für starke Maschinen! Alle Cursor- und Ctl-Befehle. 4k ROM auf der Karte für DOS, PRODOS, CP/M, PASCAL. 2 Zeichensätze verfügbar z. B. deutsch u. ASCII. Alle Features: Hoch-/Tiefstellen, autom. Unterstreichen, var. Zeichen und Zeilenabst., autom. Papierzuführung usw.

TYPETERM – ein Produkt von

interkom electronic
Kock & Mreches GmbH
Postf., 3004 Isernhagen 4
Telefon 05139-87393

Ausgabe mit TYPETERM® JUNIOR

im Slot Ihres

APPLE II/IIe/IIgs

Paketpreis jetzt **DM 799,–!**
Schreibmaschine AX-10 mit Interface TYPETERM JUNIOR, steckfertig.

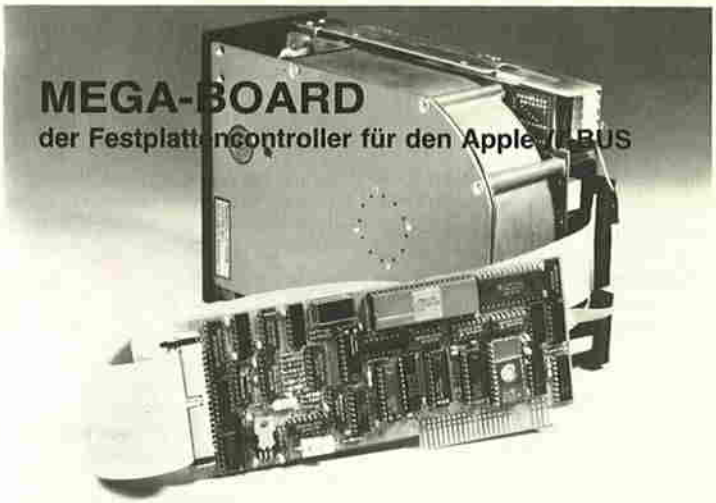
brother
Die Zukunft heute

TYPETERM JUNIOR mit AX-10 – unser besonders günstiges Gespann, ebenfalls steckfertig. Mit TYPETERM JUNIOR kann die AX-10 mehr. Sie wird zum vollwertigen Typenraddrucker für Ihren Apple:

- 3 verschiedene Schriftstärken
- Automatisches Unterstreichen
- 2 Zeichensätze z.B. deutsch u. ASCII
- 2 Zeichenabstände
- 2k ROM auf der Karte für Ausgabe unter DOS, PRODOS, CP/M u. PASCAL.

TYPETERM JUNIOR – ein Produkt von

interkom electronic
Kock & Mreches GmbH
Postf., 3004 Isernhagen 4
Telefon 05139-87393



Nicht jeder kann und möchte an seinen Apple //-Rechner unser MEGA-CORE oder die MDB 10/20 anschließen. Gerade für diesen Personenkreis bietet sich das MEGA-BOARD als der alternative und preiswerte Einstieg zum komfortablen Festplattenbetrieb an. Das Manual führt Sie zielsicher zu Ihren Wünschen:

- Festplattenbetrieb von 5-64 MBytes
- Betriebssystembereiche frei wählbar
- Booten von der Festplatte
- Betriebssysteme DOS, MS-DOS, CP/M, UCSD-Pascal, ProDOS menuegesteuert im Zugriff.

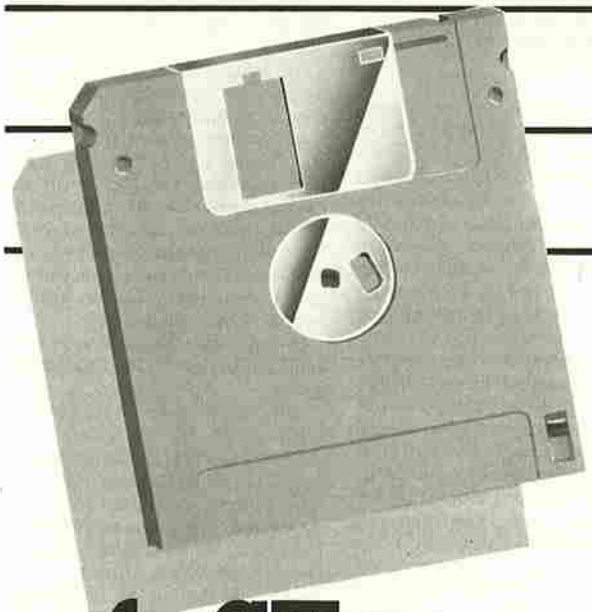
Zum Lieferumfang gehört der Controller MEGA-BOARD, alle Kabel mit Steckern, Installationssoftware, das ausführliche Manual und auf Wunsch die Festplatte zum Tagespreis.

Ein Produkt von:

FRANK & BRITTING
Elektronik Entwicklungs GmbH
Lange Straße 4, 7529 Forst
Telefon: 07251 /103068-69
Telex: 7822452 fub d

Die Harddiskcontroller-Spezialisten

Das ist eine Zeitschrift



faSTer Disk Mag
Für alle Atari ST Computer

Die Diskettenzeitschrift **FaSTer Disk Mag**, randvoll mit Programmen und Information, gibt es zum günstigen Preis:

24,80 DM

Wecken Sie die Power Ihres Atari ST mit einem Doppelklick!

Das ungewöhnliche Magazin bringt Ihnen alle zwei Monate eine Diskette prall voll mit professionell geschriebenen, lauffähigen Programmen – größtenteils sogar mit Quellcode: Anwendungen, Desk-Accessories und tolle Spiele.

Nie mehr brauchen Sie seitenlange Listings einzutippen und dann stundenlang nach Fehlern zu suchen. Mit **FaSTer Disk Mag** – dem magnetischen Magazin – sind Sie von neuen Superprogrammen nur einen Mausklick weit entfernt.

FaSTer Disk Mag bietet auch Kurse in PASCAL, C und BASIC in der englischen Muttersprache des Atari ST.

Erfahrene Autoren vermitteln Ihnen Wissen auf leichte Weise. Weiterhin lesen Sie topaktuelle Berichte über neue Hard- und Software aus Amerika, England und Deutschland. Darüber hinaus finden Sie lohnende Listen mit amerikanischer Public-Domain-Software und vieles mehr in **FaSTer Disk Mag**, dem magnetischen Magazin.

FaSTer Disk Mag erscheint alle zwei Monate.

Holen Sie sich Ihre Diskette bei Ihrem Atari-Fachhändler oder direkt vom
Dr. Alfred Hüthig Verlag
Postfach 10 28 69
6900 Heidelberg

Hüthig

Logik-Analysator

von Dipl.-Ing. Gerhard Berg

1. Was ist ein Logik-Analysator?

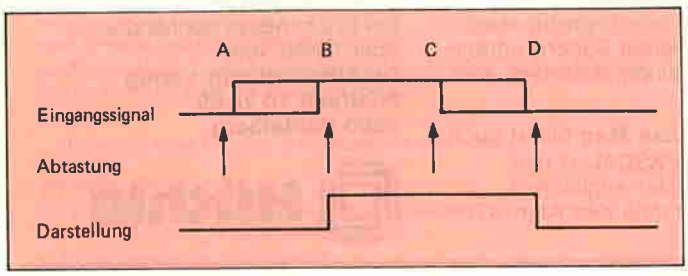
Ein Logik-Analysator (engl. Logic Analyser) ist ein Gerät zum Messen des zeitlichen Verlaufs von elektrischen Signalen. Im Gegensatz zu einem Oszillographen kann ein Logik-Analysator aber keine Analogsignale, sondern nur Digitalsignale messen, d.h. der Logik-Analysator kann keine Spannungswerte messen, sondern nur die Spannungsbereiche High und Low entsprechend den logischen Zuständen 0 und 1 unterscheiden. Folglich ist er, wie der Name schon sagt, (nur) zur (Funktions-)Analyse von logischen Schaltungen geeignet.

Ein weiterer wichtiger Unterschied zwischen einem Oszillographen und einem Logik-Analysator ist, daß der Logik-Analysator die Eingangssignale nicht kontinuierlich erfaßt und sofort ausgibt, sondern die Eingangssignale in bestimmten Zeitabständen abtastet und die gemessenen Werte zunächst zwischenspeichert und erst später ausgibt. Außerdem hat ein Logik-Analysator normalerweise mehr Kanäle (üblicherweise 8, 16, 24 oder 32), die parallel (gleichzeitig) erfaßt, gespeichert und ausgegeben werden. Die Zwischenspeicherung der Meßwerte hat den Vorteil, daß auch einmalige Vorgänge gemessen und die einmal gespeicherten Werte beliebig oft für unterschiedliche Darstellungen aus dem Speicher abgerufen werden können.

Die wichtigsten Kriterien für die Auswahl eines Logik-Analysators sind die Leistungsdaten des Zwischenspeichers, wie Speicherbreite (d.h. die Anzahl der Signale, die parallel gemessen werden können), Speicherlänge (d.h. die Anzahl der Meßwerte, die nacheinander abgetastet und gespeichert werden können) und maximale Taktfrequenz des Speichers (d.h. die maximale Frequenz, mit der die Abtastung erfolgen kann). Weitere wichtige Kriterien sind die unterschiedlichen Möglichkeiten der Takterzeugung, der Triggerung und der Ausgabe.

Unter *Takt* (engl. Clock) versteht man das Signal, mit dem die Eingangssignale abgetastet und in den Speicher geschrieben werden. Der Takt kann entweder intern oder extern erzeugt werden. Bei internem Takt erfolgt die Abtastung mit einer festen, in Stufen wählbaren Frequenz. Bei externem Takt wird das Taktsignal dem Logik-Analysator von außen zugeführt und meist dem zu analysierenden Gerät bzw. System entnommen. Bei externem Takt kann normalerweise auch noch gewählt werden, ob die Abtastung zum Zeitpunkt der ansteigenden oder abfallenden Flanke des Taktsignals erfolgt.

Unter *Triggerung* versteht man die Festlegung des Zeitpunktes, ab dem die Eingangssignale abgetastet und gespeichert werden. Die Triggerung kann entweder intern oder extern erfolgen. Bei interner Triggerung wird die Triggerbedingung aus den Datensignalen (evtl. mit Maskierungen und Verknüpfungen) abgeleitet. Bei externem Trigger wird die Triggerbedingung aus einem von außen zugeführten Signal (ansteigende oder abfallende Flanke) oder der Verknüpfung von mehreren Signalen erzeugt. Zusätzlich kann meist der Beginn der Erfassung und Speicherung noch durch die Definition einer Verzögerung (engl. delay oder displacement) verschoben werden. Durch Definition einer negativen Verzögerung können meist sogar Ereignisse vor (!) dem Triggerzeitpunkt erfaßt werden.



Die Ausgabe ist in unterschiedlichen Formen möglich. Eine Form ist das Zeitdiagramm, bei der die gemessenen Signale als Kurven untereinander dargestellt werden. Dabei besteht meist noch die Möglichkeit, die Kurven zu vergrößern, mit einem verschiebbaren Zeiger (engl. pointer oder cursor) Zeitabstände zu messen oder den ganzen Bildausschnitt zu verschieben. Als zweite Ausgabemöglichkeit können die gemessen Werte aufgelistet werden, wobei die Zahlenbasis (binär, hexadezimal usw.) noch gewählt werden kann. Diese Ausgabeform kommt in Betracht, wenn die Signale eines Adreß- oder Datenbusses eines Computers oder Peripheriegerätes gemessen wurden. Komfortablere Geräte ermöglichen zudem die Disassemblierung von Befehlsfolgen. Bei Geräten, die auf einem Computersystem basieren, kann noch die Möglichkeit bestehen, die Ergebnisse auf Diskette zu speichern oder über einen Drucker auszugeben.

Für die Messung mit dem Logik-Analysator muß noch auf ein Problem hingewiesen werden, daß sich aus der punktweisen Abtastung der Eingangssignale ergibt. Wenn nämlich der zeitliche Abstand zwischen den Abtastungen nicht kurz genug im Vergleich zu den zu messenden Impulsen ist, kann es passieren, daß Impulse ganz verlorengehen oder verfälscht dargestellt werden.

Bild 1 verdeutlicht die Zusammenhänge. Da der Zustand der Eingangssignale nur zu den definierten Abtastpunkten in den Speicher übernommen wird, kann das Eingangssignal irgendwann zwischen A und B von 0 nach 1 und irgendwann zwischen C und D von 1 nach 0 gehen, ohne daß sich die Darstellung ändert. Bei einem Abtastabstand T und dem dargestellten Impuls der Länge 2 · T kann das Eingangssignal die Länge 1 · T bis 3 · T haben, was einem Fehler von max. +/- 50% entspricht. Um einen akzeptablen Fehler (max. +/- 10%) zu erzielen, darf entsprechend der Abtastabstand höchstens 1/10 der Impulsbreite sein bzw. die Abtastfrequenz muß mindestens 10mal höher als die zu messende Frequenz sein. Um sicher zu gehen, daß alle Impulse überhaupt dargestellt werden und kein Impuls verlorengeht, muß die Abtastfrequenz mindestens doppelt so hoch wie die zu messende Frequenz sein.

Etwas anders liegen die Verhältnisse, wenn mit externem Takt die Adressen oder Daten auf einem Bus erfaßt werden sollen. Da hierbei nur die Werte an den Abtastpunkten (beim 6502 z.B. an der abfallenden Flanke von Φ0) und nicht die dazwischenliegenden Werte interessieren, reicht es hier, wenn die Abtastfrequenz gleich der Nutzfrequenz ist.

2. Test des Logik-Analysators Axelen ALA0810

2.1. Bestandteile

Der Logik-Analysator ALA0810 von Axelen besteht aus folgenden Komponenten:

– Einer Steckkarte, die in einen beliebigen Steckplatz (empfohlen 2 bis 5) des Apple II gesteckt werden kann. Diese Karte enthält nur zwei ICs und stellt lediglich die Verbindung zum eigentlichen Logik-Analysator her.
– Einem Metallgehäuse (27 x 21 x 4 cm), das den eigentlichen Logik-Analysator beinhaltet. Der Logik-Analysator hat kein eigenes Netzteil, sondern deckt seinen gesamten Strombedarf (+5V, 1,5A) aus dem Apple-Netzteil. Bei vollbestückten Systemen müssen evtl. andere Karten entfernt werden, um das Netzteil nicht zu überlasten.
– Einem Plastikgehäuse (6 x 10 x 2,5 cm), aus dem die 11 Testleitungen abgehen.
– Den notwendigen Verbindungskabeln.
– Einem 56seitigen Handbuch (Version 3.0). Das Handbuch liegt nur in englischer Sprache (und dazu noch in sehr schlechtem Englisch und mit vielen Tippfehlern) vor. Eine Befehlsübersicht erleichtert jedoch das Arbeiten.
– Zwei Disketten mit der Betriebssoftware (Version 3.1). Die Disketten trugen die gleiche Bezeichnung, sollen aber nach Angaben der Vertriebsfirma unterschiedlich sein. Da eine der Disketten beim Autor nicht gelesen werden konnte, konnten evtl. Unterschiede nicht untersucht werden.

Das Ganze ist gut verpackt geliefert worden. Eine zusätzliche, einer Kollegmappe ähnliche Tasche, dient zur Aufbewahrung der Komponenten.

2.2. Funktionen

Der Logik-Analysator ALA0810 bietet folgende Funktionen:

– 8 Kanäle, 4096 Meßwerte pro Kanal.
– Interner Takt mit 16 wählbaren Taktfrequenzen von 10 ms (100 Hz) bis 100 ns (10 MHz).
– Externer Takt bis max. 10 MHz mit wählbarer Flanke und 16 wählbaren Teilerstufen zwischen 1 und 100000.
– Interner Trigger mit bis zu 3 durch ODER verknüpften Masken. Innerhalb jeder Maske kann jedes Bit als 0, 1 oder beliebig (don't care) definiert werden.
– Externer Trigger (Unconditional = unbedingter Trigger).
– Trigger-Verzögerung wählbar zwischen -4095 und 0 Takten.
– Ausgabe als Impulsdiagramm. Das Diagramm kann in 6 Stufen von 1facher bis 32facher Vergrößerung dargestellt werden. Je nach Vergrößerung enthält ein Bildschirmfenster 8 bis 256 Meßpunkte. Über einen Zeit-Zeiger und -Zähler können Zeitdifferenzen einfach gemessen werden.
– Ausgabe als Liste. Jedes der acht Signale kann einer von zwei Gruppen zugeordnet oder gar nicht ausgegeben werden. Für jede Gruppe kann ein Kennzeichen (1 Buchstabe), die Zahlenbasis (binär, oktal, dezimal oder hexadezimal) und die Polarität (normal oder invertiert) gewählt werden.
– Die erfaßten Daten können auf Diskette gespeichert und zur späteren Auswertung wieder eingelesen werden.
– Alle Ausgaben, die auf dem Bildschirm erfolgen, können auch auf einem Drucker ausgegeben werden.

2.3. Test

Zum Test wurde der Logik-Analysator in einem Apple-II+-kompatiblen Rechner installiert. Gemessen wurden die Signale in einem zweiten Rechner (einem ebenfalls Apple-II+-kompatiblen ITT 2020). Mit entsprechenden Englischkenntnissen und Erfahrung in der Benutzung von Logik-Analysatoren war es einfach, die Bedienung zu verstehen. Nach einmaligem Durchlesen des Handbuchs reichte die zweiseitige Befehlszusammenfassung für die Bedienung völlig aus.

Erste Schwierigkeiten gab es, als jeder Versuch scheiterte, etwas auszudrukken. Das Handbuch empfiehlt („recommends") Epson-kompatible Drucker. Der Autor benutzt einen Epson LX-80 allerdings mit einer selbstentwickelten Interfacekarte. Die Software der Karte hält sich streng an die Konventionen der Ausgabekarten. Es muß deshalb vermutet werden, daß die Ausgaberoutinen des Logik-Analysators (selbst bei Textausgabe) nicht über die „offiziellen" Ausgabevektoren arbeiten, sondern direkt auf die Hardwareregister der Drukker-Interfacekarte zugreifen. Das Handbuch enthält keine Informationen über die Anpassung von nicht Epson-kompatiblen Druckern oder Interfacekarten.

Für den eigentlichen Test wurden zwei praxisnahe Messungen gemacht. Bei der ersten Messung wurden die Taktsignale des Apple-Busses gemessen: 7M (8,8 MHz beim ITT), Q3, Φ0, Φ1, R/W usw. Dabei wurde der interne Takt mit 10 MHz Taktfrequenz benutzt. Die Messung und alle Ausgabefunktionen funktionierten einwandfrei. Die Darstellung der Impulsdiagramme im HGR-Modus des Apple ist übersichtlich und hat eine ausreichende Auflösung. Das 7M-Signal konnte wegen des Meßprinzips und der im Vergleich zu niedrigen Abtastfrequenz (siehe Teil 1 dieses Beitrages) natürlich nicht richtig wiedergegeben werden.

Im zweiten Test wurden die acht Datenbus-Signale gemessen und extern auf die negative Flanke von Φ0 getaktet. Ab Speicherplatz $0300 wurde das „Programm" JMP $0300 = 4C 00 03 eingegeben und gestartet. Fehlerhafterweise gingen bei diesem Test die meisten „00"-Bytes verloren. Wurde das Programm auf andere Stellen verschoben, so daß aus dem „00" andere Werte wurden, so wurden die Verluste geringer. Da keine Schaltbilder des Logik-Analysators zur Verfügung standen, konnte dieser Fehler nicht näher untersucht werden. Es kann auch nicht gesagt werden, ob es sich um einen Design-Fehler oder um einen Fehler des einzelnen Gerätes handelt.

Mit den gleichen Anschlüssen sollte dann der Start der Software nach dem Drücken der Reset-Taste erfaßt werden, wie dies gerne zur Fehlersuche bei defekten Rechnern gemacht wird. Dies war jedoch nicht möglich, da eine Umschaltung zwischen internem und externem Trigger fehlt und bei dem vorhandenen externen („unconditional") Trigger die Flanke nicht umgeschaltet werden kann.

Der Verlust der auf Diskette aufgezeichneten Meßergebnisse war auf einen Bedienungsfehler des Autors zurückzuführen. Da dies jedoch in der „Hitze des Gefechts" leicht passieren kann, sollte hier besonders darauf hingewiesen werden. Beim Abspeichern von Meßergebnissen (und dem Löschen von Dateien) wird das Inhaltsverzeichnis (Directory) auf der Diskette nicht gleich auf den neuesten Stand gebracht, sondern erst wenn das Programm mit einem speziellen Befehl verlassen wird. Wird das Programm ohne diesen Befehl abgebrochen, so sind nach erneutem Booten evtl. alle auf der Diskette gespeicherten Meßergebnisse verloren. Schließlich können Meßergebnisse, die auf Laufwerk 2 abgespeichert werden, nach erneutem Booten überhaupt nicht mehr geladen werden.

Neben diesen gravierenden Problemen sind beim Test eine Reihe von Punkten aufgefallen, die zwar die Funktionsfähigkeit nicht unmittelbar beeinträchtigen, aber die Benutzung unnötig erschweren.

– Alle Fehlermeldungen im Format-Modus blinken nur kurz auf und werden sofort wieder gelöscht.
– Der Zeitzeiger in den Impulsdiagrammen besteht nur aus einem Pfeil unter der untersten Kurve und ist keine Linie, die durch alle Kurven geht. Ein genaues Ablesen der obersten Kurve ist deshalb erst bei achtfacher Vergrößerung vernünftig möglich. Dabei werden jedoch nur noch 32 Meßpunkte im Bildschirmfenster dargestellt.
– Von der Ausgabe der Impulsdiagramme oder einer Liste kann die nächste Messung nicht direkt gestartet werden. Man muß vielmehr erst unnötigerweise durch den „Format-Modus".
– Bei internem Takt erfolgt ab 200 µs Taktfrequenz die Meldung „Trigger not found", auch wenn ein Trigger gefunden wurde.

2.4. Wertung

Dem Werbeslogan des Herstellers „Axelen is excellent" kann für dieses Produkt leider (noch?) nicht zugestimmt werden. Alle Komponenten (Hardware, Software und Dokumentation) machen einen noch nicht ausgereiften, professionellen Ansprüchen nicht voll genügenden Eindruck. Vor einem möglichen Kauf sollte man genau prüfen, ob die Leistungsdaten für die eigene Anwendung ausreichen und sich vom Lieferanten einwandfreie Funktion und Kompatibilität mit dem eigenen Drucker und Interface garantieren lassen.

SUPERQUICK

Ein superschnelles Disketten-Kopierprogramm

von Arne Schäpers, 1985, Programmdiskette mit Anleitung, DM 38,–

Mit SUPERQUICK ist es möglich, Disketten jeden Formats (DOS 3.3, ProDOS, UCSD-Pascal und CP/M) in einer unglaublich kurzen Zeit von nur 29 Sekunden (mit Formatierung) zu kopieren. Bei entsprechender Speichererweiterung kann der gesamte Disketteninhalt eingelesen werden, um mehrere Kopien anzufertigen. Die Zeit für eine Einzelkopie reduziert sich dann auf sage und schreibe 19 Sekunden.

SUPERQUICK erkennt die 64K-Karte (in Slot 3) des Apple IIe und IIc sowie eine 16K-Language-Card in Slot 0 und bezieht diese selbständig als Datenpuffer ein. Darüber hinaus werden die IBS-Karten AP17 in den Ausbaustufen 64K bis 256K automatisch unterstützt und gegebenenfalls als weitere Puffer eingesetzt.

Jetzt mit Spezialprogramm für 160-Spur-Erphi-Laufwerke!

Hüthig Software Service · Postfach 10 28 69 · 6900 Heidelberg 1

Aztec-C-Compiler

Ein Erfahrungsbericht

von Dipl.-Ing. K.-W. Bott

Mit dem Aztec-C-Compiler-Paket des Herstellers Manx Software Systems ist ein leistungsfähiges C-Entwicklungspaket für Computer der Apple-II-Serie verfügbar. Dem Testbericht liegt der Aztec-C-Compiler in der Version 1.05 zugrunde. Aufgrund des Umfangs der Software mußte diese auf mehrere Disketten im DOS-3.3-Format verteilt werden. Aztec-C erfordert einen Apple IIe/c/+ mit 64K RAM und DOS 3.3. Im folgenden werden die wichtigsten Module des Paketes beschrieben.

1. Shell

Nach dem Booten der Startdiskette befindet man sich in der sog. Shell, einer dem Betriebssystem UNIX nachempfundenen Benutzeroberfläche. Zuerst sucht die Shell auf der Diskette ein Programm mit dem Namen „.profile", vergleichbar mit dem DOS-3.3-Hello-Programm, das als erstes ausgeführt wird. Die Shell ist eigentlich ein Kommando-Interpreter ähnlich dem CCP unter CP/M, der die Interpretation der eingegebenen Befehle übernimmt. Eine Liste der wichtigsten Befehle, die die Shell interpretiert, sind in der **Tabelle 1** zusammengestellt.
Eine besondere Eigenschaft der Shell ist die Verwaltung externer Geräte wie Drukker und Tastatur. Mit dem Symbol „>" kann eine Ausgabe, die normalerweise auf dem Bildschirm erfolgt, auf ein anderes Gerät gelenkt werden. Außerdem ist es möglich, die Standardeingabe über die Tastatur durch den Inhalt einer Datei zu ersetzen. Einige Beispiele hierfür sind
„cat textdatei >pr:":
Dieser Befehl gibt den Inhalt der Datei „Textdatei" auf dem Drucker aus.
„cmd <PARM":
Das Programm „cmd" nimmt seine Eingabeparameter aus der Datei PARM.

Neben den implementierten Kommandos aus Tabelle 1 können weitere Dienstprogramme auf Diskette abgelegt werden. Dies sind entweder eigene, mit dem C-Compiler erstellte, oder bereits vorhandene Programme wie z.B OD oder NM, auf die später eingegangen wird. Programme dieser Art werden durch den Aufruf „Programmname Argument1 Argument2 ..." in den Arbeitsspeicher geladen und ausgeführt. Als zweite Möglichkeit können sog. „Batch-Dateien", ähnlich den CP/M-SUBMIT-Dateien, verarbeitet werden. Hat z.B. die Textdatei „batch" den Inhalt „cp $1,s6,d1 $2,s6,d2", wird durch den Aufruf „batch name1,name2" die Datei mit „name1" von Laufwerk 1 in die Datei mit „name2" nach Laufwerk 2 kopiert.
Die Benutzeroberfläche der Shell ist etwas gewöhnungsbedürftig, jedoch für ein Entwicklungspaket sinnvoll. Zu bemängeln sind die Fehlermeldungen, die allzuoft ausbleiben. Gibt man zum Beispiel „cd s27" ein, erhält man keine Fehlermeldung, und das System verhält sich so, als wäre eine korrekte Einstellung des Datenlaufwerks vorgenommen worden.
Die Shell belegt Bank 2 und teilweise Bank 1 der Language-Card, so daß dem Anwender für eigene Programme der Bereich $0800-$9600 zur Verfügung steht. Nach dem Booten der Startdiskette wird das DOS 3.3 durch Starten der Shell an verschiedenen Stellen „gepatcht".

2. C-Compiler

Das Programmpaket enthält zwei C-Compiler in Gestalt der Dateien C65, C65.2 bzw. CCI. Der Sprachumfang entspricht dem Standard, der in dem Buch „Programmieren in C" von Dennis M. Ritchie, einem Mitentwickler der Sprache C, und Brian W. Kernighan definiert wird. Einziger Unterschied ist das Fehlen von Bitfeldern, die nicht zum Sprachumfang des C-Compiler-Pakets gehören.
Der C-Compiler C65 erzeugt einen 6502-Assembler-Quelltext, während CCI einen Quelltext erzeugt, der dem einer virtuellen Maschine entspricht und während der Programmausführung von einem Interpreter umgesetzt wird. Die wichtigsten Compiler-Optionen sind in **Tabelle 2** aufgeführt. Hier wird fast alles geboten, was zu einer effektiven Programmentwicklung nötig ist. Wird das erstellte C-Programm größer als der verfügbare freie Arbeitsspeicher, können Overlays generiert werden. Ein Beispiel hierzu findet sich in einem der zahlreichen Archive, auf die später eingegangen wird.
Nach Aufruf des C-Compilers wird zunächst der C-Preprozessor gestartet, um einen Textersatz der „#"-Statements vorzunehmem und die Code-Generierung zu steuern. Der Preprozessor ist gegenüber dem Standard erweitert, so daß mit den Preprozessor-Befehlen „#asm" bzw. „#endasm" direkt 6502-Quelltexte in den C-Quelltext eingebunden werden können. Allerdings wird dadurch die Portabilität des C-Programms stark eingeschränkt, da ein Fremdcompiler einen solchen Quelltext wahrscheinlich nicht übersetzen kann.
Nach einem erfolgreichen Preprozessorlauf wird bei einer korrekten Syntax des C-Quellprogramms ein 6502-Assembler-Quellfile erzeugt. Findet der Übersetzer einen Fehler, wird die Zeilennummer ausgegeben, in der der Fehler auftrat, sowie eine Fehlernummer oder eine Fehlermeldung im Klartext.
Hier traten im Test des Compilermoduls C65 bzw. C65.2 unter bestimmten Bedingungen Abstürze des Compilers auf. Nach vielfacher Ausgabe der gleichen Fehlermeldung findet man sich dann im Monitor wieder, den man am besten mit Ctrl-Reset wieder verläßt.

3. AS65 und ASI

Dem Programmpaket sind zwei Assembler beigefügt. AS65 übersetzt den 6502-Quelltext in einen relokatiblen Objektcode, während ASI als Quelldatei einen Assembler-Quelltext der virtuellen Maschine benötigt. Die Assembler AS65 bzw. ASI werden direkt am Ende eines erfolgreichen Compilerdurchlaufs aufgerufen, sofern keine entsprechende Option gesetzt wurde.

4. LN

Der Linker LN bindet relokatible Programme, die aus assembliertem Quelltext entstehen, zusammen. Dabei können Funktionen aus den zahlreichen Bibliotheken angehängt werden. Auch hier stehen Optionen zur Verfügung (Tabelle 2), die den Bindevorgang steuern.
Daten- und Programmbereich können durch die Schalter „-c" und „-d" festgelegt werden, was das Erzeugen eines ROM-fähigen Codes ermöglicht.

5. Mitgelieferte Tools

VED: Der Programm-Editor VED ist sicherlich einer der Schwachpunkte des Programmpaketes. Es werden nur minimale Hilfen zum Erstellen und Ändern eines Programmtextes geboten. Der VED ist vollständig in der Programmiersprache C geschrieben; der Quelltext befindet sich in einem der Archive, so daß man den VED nach eigenen Wünschen modifizieren kann.
ARCH: Das Programm ARCH dient dazu, die zahlreichen vorhandenen Archive zu nutzen und eigene zu erstellen. Fast alle Quelltexte der Library-Funktionen und mitgelieferten Dienstprogramme findet man in den Archiven wieder. ARCH ermöglicht eine komprimierte Form des Abspeicherns von Quelltexten. ARCH selbst ist ebenfalls als C-Quelltext in einem Archiv vorhanden. Eine Zusammenstellung der wichtigsten Funktionen findet man in **Tabelle 3**.
MKLIB: Das Programm MKLIB repräsentiert den Library-Manager des Systems. Mit Hilfe von MKLIB können neue Bibliotheken erstellt, Programm-Module zu einer bestehenden Bibliothek hinzugefügt oder extrahiert werden. Die Bibliotheken können beim Linken angegeben werden, so daß der Linker automatisch nach den extern oder nicht definierten Funktionen in den Libraries sucht und sie zu dem Objektfile hinzufügt.
Zum Lieferumfang gehören die Bibliotheken „xx65.LIB" für Programme, die mit C65 erstellt wurden und unter der Shell ablaufen. „xxINT.LIB" wird von Programmen benötigt, die von CCI erzeugt werden, während „SAxx.LIB" für Programme vorgesehen ist, die nicht mit der Shell, sondern direkt unter DOS 3.3 laufen sollen.
OD: Ein Programm zur Ausgabe eines Files im Hex- und ASCII-Format.
CMP: Vergleicht zwei Dateien. Je nach Option werden alle unterschiedlichen Bytes ausgegeben oder nach dem ersten ungleichen Byte abgebrochen.
NM: Gibt eine Symboltabelle eines relokatiblen Objektcodes aus.

Tabelle 1
Aztec-C-Shell-Kommandos

ls	Inhaltsverzeichnis	ls [sl,d,vl]
mv	Umbenennen von Dateien	mv filel [s,d,v] file2 [s,d,v]..
rm	Löschen von Dateien	rm filel [s,d,v] file2 [s,d,v] ..
cp	Kopieren von Dateien	cp filel [s,d,v] file2 [s,d,v]
cat	Inhalt von Dateien listen	cat filel [s,d,v] file2 [s,d,v] ..
cd	Einstellung des Datenlaufwerks	cd s,d,v
ce	Ändern des Programmlaufwerks	ce s,d,v
call	Aufruf eines Prog. ab Adresse	call [$]nnnn
run	Start eines Prog. im Speicher	run [parml] [parm2] ..argl
load	Laden einer Datei von Diskette	load file [s,d,v] a$nnnn l$mmmm
save	Speicherbereich auf Diskette	save file [s,d,v] a$nnnn l$mmmm
maxfiles	Anzahl der File-Puffer	maxfiles nr
bye	Sprung in den Monitor	bye
boot	Kaltstart von Slot n	boot n
lock	Schreibschutz für Datei	lock filename
unlock	Schreibschutz löschen	unlock filename

Tabelle 2
C-Compiler-Optionen

-o filename	Relokatibles Objektfile mit filename.rel erzeugen
-a	Assembler-Quellfile mit filename.asm erzeugen
-v	Compiler-Statistik ausgeben
-znnnn	Reservierter Bereich für Stringtabelle
-xnnnn	Reservierter Bereich für Macros
-ynnnn	Reservierter Bereich für switch/case-Verarbeitung
-ennnn	Reservierter Bereich für arithmetische Ausdrücke
-t	C-Text im Assemblerlisting

Linker Optionen

-v	Linker-Statistik ausgeben
-cnnnn	Adresse des Programmbereichs ändern
-dnnnn	Adresse des Datenbereichs ändern
-r	Ausgabedatei.rsm für neues Linken
-o filename	Name des Objektfiles ändern
-f file	Kommandodatei mit Linkeranweisungen
-l NAME	Bibliothek mit "NAME" verwenden

Assembler Optionen

-o filename	Output-file mit filename.rel
-l filename	Listfile des assemblierten Programms erzeugen
-ZAP	Löschen des Quellfiles nach Assemblierung

6. Testprogramme

Alle Testprogramme wurden auf einem Apple IIe mit erweiterter 80-Zeichenkarte, DOS 3.3 und zwei Diskettenlaufwerken getestet. Um ein C-Quellprogramm zu übersetzen, zu linken und auszuführen, müssen der Compiler, der Linker, der Assembler, eine Library und ein Quelltext auf Diskette vorhanden sein. Compiler, Linker, Assembler und Library belegen allein etwa 132K Diskettenspeicherplatz, so daß ein Arbeiten mit dem Aztec-C-Compiler nur mit zwei Diskettenlaufwerken sinnvoll ist.
Übersetzt und bindet man das Programm PRG1, fallen zwei Tatsachen besonders auf. Das ausführbare Programm PRG1 belegt etwa 5,5K Speicher. Die Ursache hierfür liegt in der Verwendung der „printf"-Funktion, die aus einer Library entnommen wird und selbst größtenteils in C geschrieben ist. Fast alle Standardfunktionen nach K&R sind in Bibliotheken vorhanden, führen jedoch zu einem umfangreichen Objektcode, der die Größe des verfügbaren Arbeitsspeichers stark herabsetzt, besonders wenn diese in C geschrieben wurden. Dieses Phänomen findet man auch bei anderen C-Compilern, z.B. dem C-Compiler des Atari-Entwicklungspakets, nur steht auf modernen PCs ein Mehrfaches von 64K als Speicherplatz zur Verfügung.
Abhilfe schafft das Programm PRG2. Es erzeugt einen Objektcode von etwa 3K Länge, leistet aber das gleiche wie Programm PRG1. PRG2 verwendet die Funktion „putchar" zur Ausgabe der Zeichenkette. „putchar" ist zwar ebenfalls über Library-Funktionen definiert, ist aber in 6502-Assembler realisiert und gibt ein Zeichen aus. Der gesamte Overhead der „printf"-Funktion bezüglich der formatierten Text- und Zahlendarstellung entfällt. Nur durch eine effektive Programmierung und durch Studium der Implementierungen der Standardfunktionen kann das Aztec-Paket den relativ kleinen Programmspeicher des Apple II optimal nutzen, so daß auch größere Anwendungen realisiert werden können.

Weiterhin fällt eine lange Übersetzungs- und Linkzeit auf. Bei größeren Programmen können Sie getrost eine Kaffeepause einlegen. Die Ursache ist hauptsächlich in den langsamen Diskettenzugriffen zu suchen, da alle Module von Diskette gelesen werden müssen. Temporäre Dateien, die der Compiler während der Übersetzungsphase erzeugt, werden ebenfalls auf Diskette abgelegt. Empfehlenswert ist die Installation eines DOS-3.3-RAM-Disk-Drivers, der beispielsweise auf der „MMU 2.0"-Diskette (Hüthig Software Service) enthalten ist.
Als weiteres Testprogramm wurde die Akkermann-Funktion gewählt (PRG3). „Berüchtigt" wegen ihrer Rekursivität, ist sie ein guter Test auf eine leistungsfähige Stackverwaltung. Erst bei Ackermann (3,4) verabschiedet sich das Programm mit einem Sprung in den Apple-Monitor.
Als abschließender Test wurden die Laufzeiten des Programm PRG4 in verschiedenen Versionen verglichen. In der ersten Version wurden alle Variablen der Klasse Integer zugeordnet. In der zweiten sind die Variablen als „register integer" deklariert. Die Laufzeit der zweiten Version war um die Hälfte geringer als die der ersten Version. „register"-Variablen werden mangels geeigneter 6502-CPU-Register durch direkte Zero-Page-Adressierung realisiert, während andere Variablen immer durch indirekt-indizierte Adressierung (Stack) verarbeitet werden.

7. Zusammenfassung

Der Erwerb des Aztec-C-Compiler-Paketes für den Apple ist eine durchaus empfehlenswerte Investition. Die C-Compiler stellen den fast vollständigen K&R-Sprachumfang auf dem Apple zur Verfügung. Nahezu alle Implementierungen der Bibliotheksfunktionen sowie der Dienstprogramme sind als Quellcode in C oder 6502-Assembler auf den mitgelieferten Disketten vorhanden. Daraus resultiert eine Dokumentation des Programmpakets, die als außergewöhnlich bezeichnet werden kann. Nicht nur dem Anfänger, der die Sprache C erlernen möchte, bietet das Programmpaket aufgrund des fast vollständigen Sprachumfanges einiges, auch Programmierer mit höheren Ambitionen können dank der sehr guten Dokumentation dem etwas antiquierten Apple II etwas abgewinnen.
Der positiven Gesamteindruck wird durch den nicht ganz absturzsicheren Compiler C65, den etwas zu spartanisch ausgestatteten Text-Editor VED sowie die relativ langen Übersetzungs- und Linkzeiten getrübt, die aber eher die Grenzen eines Apple-II-Systems mit 6502-CPU aufzeigen.

Tabelle 3
Optionen des Archivers

-x name	Eine Datei aus dem Archiv entfernen
-a name	Eine Datei an ein bestehendes Archiv anhängen
-l	Inhaltsverzeichnis ausgeben
-c	Inhalt eines Archivs löschen

Optionen des Library-Managers

-t	Inhaltsverzeichnis der Bibliothek ausgeben
-x	Modul aus der Bibliothek extrahieren
-r	Modul in der Bibliothek ersetzen
-a	Modul in bestehende Bibliothek aufnehmen

```
/*
 * Testprogramm PRG1
 */
main()
{
        printf("Testprogramm #1 - Aztec C-Compiler\n");
}

/*
 * Testprogramm PRG2
 */
main()
{
        char *ausgabe = "Testprogramm #2 - Aztec C-Compiler\n";

        while(*ausgabe != '\n') {
                putchar(*ausgabe);
                ausgabe++;
        }
        putchar(*ausgabe);
}

/*
 * Testprogramm PRG3
 * Ackermann-Funktion
 */
main()
{
        int x,y;

        for(x=0; x<10;x++)
            for(y=0; y<10;y++)
                        printf("ackermann(%d,%d) = %d\n",x,y,acker(x,y));

}
acker(x,y)
int x,y;
{
        if(!x)
             return(y+1);
        else {
             if(!y)
                return(acker(x-1,1));
        else
                return(acker(x-1, acker(x, y-1)));
        }
}

/*
 * Testprogramm PRG4
 * Version 1: Variablen Integer
 * Version 2: Variablen register integer
 */
#define BELL 0x07
main()
{
        int var1, var2;        /* für Version 2: register int var1, var2; */
        char c;

        var1 = var2 = 30000;

        while((c = getchar()) != ' ') {;} /* zum Start Leertaste drücken */
        while( var1)
          var1--;
        while( var2)
          var2--;
        putchar(BELL);                    /* Testende mit Piepser anzeigen */
}
```

Apple Software für Vertreter und Reisende

Terminverwaltung, Berichtsprogramm, Textbearbeitung

von K. W. Hillerkus

1986, Diskette mit Manual,
DM 278,—
ISBN 3-7785-1322-2

Die auf dieser Diskette enthaltenen Programme sind auf ein Vertreter-/Reisenden-Büro zugeschnitten und sollen die immer wiederkehrenden Arbeiten erleichtern oder ganz übernehmen, wie z. B.:

- Berichte schreiben mit automatischer Terminverwaltung.
- Termine/Daten/Namen usw. auch kombiniert suchen und auslisten.
- Anschriften verwalten und automatisch einem Bericht zufügen.
- Tagestermine verwalten und auf Datum oder Tagesname ausgeben.
- Kundentermine suchen, korrigieren, kopieren, ändern, ausgeben.
- Während der Berichtserstattung neue Kunden aufnehmen.
- Auf die Berichte einen verschlüsselten Termin eintragen.
- Zählen, wieviel Berichte ein vertretenes Werk erhalten hat.
- Briefe schreiben, mit und ohne Anschrift abspeichern, korrigieren, Zeilen zufügen oder streichen, Adresse ändern, Text am Bildschirm auslisten.
- Rundschreiben erstellen mit Anschriftenverwaltung, Anschriften werden am Bildschirm gezeigt und können angenommen oder abgelehnt werden. Auflisten, an welche Anschriften der Text ging. Adressen-Dateien ausdrucken mit Speicherbezeichnung. Adressendatei einrichten, ergänzen, korrigieren, am Bildschirm auslisten. Etiketten drucken für die angeschriebenen Anschriften oder an ausgesuchte Adressen aus der Hauptdatei.

Gerätevoraussetzung: Apple II mit CP/M-Karte oder Basis 108; MBASIC unter CP/M 2.2.

Dr. Alfred Hüthig Verlag
Postfach 10 28 69
6900 Heidelberg

Semjan presents...

• //+,e,gs CHAMPION Drucker Karten

- CIRTECH Parallele Text- u. Graphik-Druckerkarte komplett mit Kabel.
- Option 64K RAM Zwischenspeicher, 40/80 Zeichen Dump zu jeder Zeit.
- Umfangreiche Kontroll-Steuerbefehle z. B. Zeichensatzwahl, etc.
- Einsatz von DOS 3.3, PRODOS (Appleworks), UCSD PASCAL, CP/M.
- Champion-Karte voll Graphik fähig, Apple //e Graphik.
- Serieller Ausbau möglich, 75 bis 9600 Baud Ein- bzw. Ausgabe.
- Mischbetrieb, parallel und seriell, zu jeder Zeit möglich.
- Drucker: Epson, Centronics, Brother, Itoh, Imagewriter I/II, etc.

K030 //+,e,gs CHAMPION Interface DM 199,00
K031 //+,e,gs CHAMPION für Imagewriter I/II DM 259,00
K033 //+,e,gs CHAMPION Interface mit 64K RAM DM 398,00

• //+,e,c Uni-Mate UniDisk Software

- Einsatz der UniDisk Laufwerke ab sofort mit DOS 3.3, CP/M 2.20
- CP/M 2.23, PASCAL 1.1 und PASCAL 1.2. Einmalige Installation.
- Einsatz von UniDisk und DISKII Laufwerken möglich.
- DOS 3.3 bootbar, Speicherkapazität 800K, INIT möglich.
- CP/M 2.2 und 2.23 bietet 768K Speicherk.. Formatierung möglich.
- PASCAL bietet 768K Speicherkapazität. Formatierung möglich.

K045 //+,e,c CIRTECH Uni-Mate Software DM 99,00

• //+,e,gs CIRTECH EPROM System

- Super Eprom Programmier-System. System komplett mit aller Software.
- Einsatz als Text/Grafik Interface und Eprom-Programmier-System.
- Einsatz als zweifache I/O Karte möglich. 12,5 V und 21 V Betrieb!
- Einsatz von EPROM: 2716, 2732 2732A, 2764, 27128 und 27256.

K080 //+,e,gs Eprom Programmier-System DM 399,00

Händleranfragen willkommen. Neuen Katalog 4/87 anfordern.
Wir bieten auf alle CIRTECH-Produkte 12 Monate Garantie.

M. Semjan Computer Systeme

Postfach 90 01 64, 6000 Frankfurt/Main 90, Tel. 069-70 18 53
Telex 051 933 521 dmbox g. Ref: Box: DM3:SEMCOM, Mailbox-Adresse: DM3 SEMCOM

Semjan presents...

• //+,e,gs CIRTECH IMB RAM Karte

- Sehr schneller Datenzugriff, 50K/sec.. Max. 6MB RAM.
- In jedem Slot einsatzfähig. Kein 'patchen' notwendig.
- Fehler-Selbsttest. Karte entspricht Apple Standard.
- Voll DOS 3.3, Diversi-DOS, ProDOS, PASCAL 1.1, 1.2 und 1.3,
- sowie CP/M 2.2, 2.23 und CP/M 3.0 fähig.
- Laufwerke: AppleII, UniDisk, ProFile, Erphi-AFDC2/3, etc.
- Einsatz aller Apple II+,e,gs Software möglich.
- Bis zu 1012K RAM Arbeitsspeicher, z. B. APPLEWORKS 1.3/2.0,
- Pinpoint, Mouse & PFS und andere neuere Programme.
- AW 1.3 Update kostenlos, für alle AW 1.2 Besitzer.
- FLIPPER Karte wird mit Programm Manager (FPM) geliefert.
- FPM bietet: Einsatz von mehrerer Betriebssysteme möglich.
- Aufteilen der FLIPPER Karte in bis zu 4 Arbeitsbereiche.
- Automatisches laden und abspeichern der Arbeitsbereiche.

K070 //+,e,gs FLIPPER mit 1MB RAM u. FPM DM 749,00
K072 //+,e,gs plus RAM mit 256K RAM o. FPM DM 479,00

• //e,c CP/M Plus System

- CIRTECH CP/M Plus Modul belegt keinen Slot im Apple //e,c.
- Komplettes Betriebssystem CP/M 3.0 von Digital Research.
- Z80H mit 8MHz, Einsatz von 128K RAM, Drucker-Spooler mit 12K RAM.
- Laufwerke: DiskII, UniDisk, //e auch ProFile, Erphi-AFDC 2/3.
- Kompatibel zu CP/M 2.20 und 2.23. Apple //c mit Maus-Funktion.
- Einsatz von WORDSTAR, dBASE, TURBO PASCAL, GBASIC, etc.

K010 //c CP/M Plus System DM 544,00
K011 //c WORDSTAR/MAILMERGE und K010 DM 699,00
K012 //e CP/M Plus System DM 499,00
K016 //e TURBO PASCAL 3.0 und K012 DM 679,00
K019 //c CP/M Modul 2.20/2.23 o. Betr. Sys. DM 348,00
K021 //e CP/M Karte 2.20/2.23 o. Betr. Syst. DM 188,00

Händleranfragen willkommen. Neuen Katalog 4/87 anfordern.
Wir bieten auf alle CIRTECH-Produkte 12 Monate Garantie.

M. Semjan Computer Systeme

Postfach 90 01 64, 6000 Frankfurt/Main 90, Tel. 069-70 18 53
Telex 051 933 521 dmbox g. Ref: Box: DM3:SEMCOM, Mailbox-Adresse: DM3 SEMCOM

Print Shop

Ein Programmpaket für Matrixdrucker

von Dieter Charchot

1. Vielseitiger Einsatz (?)

Matrixdrucker sind die am häufigsten benutzten Peripheriegeräte an Computern, und doch werden ihre Fähigkeiten oft bei weitem nicht ausgenutzt. Für Apple-IIe-Besitzer kann sich das ändern.
Die fantastischen Möglichkeiten, die Matrixdrucker heute im Einsatz zusammen mit Personal-Computern anbieten, waren noch vor einigen Jahren weitaus teureren Anlagen vorbehalten; diese Entwicklung geht noch weiter. Unterschiedlichste Schriftarten und vor allem die Fähigkeit, grafische Elemente zu generieren, gehören inzwischen zum Standard. Und doch werden leider allzu viele dieser Allroundgeräte nur zu einfachsten Zwecken eingesetzt. Da quellen Listings oder Briefe aus den Schlitzen, gelegentliche Biorhythmuskurven oder Hardcopies, also die Wiedergabe des Bildschirminhalts auf Papier, sind schon etwas Besonderes, von anderen Zwecken ganz zu schweigen.
Eine Ursache dürfte darin liegen, daß die Ansteuerung der Grafikfunktionen zuweilen recht kompliziert ist, wenn man nur auf die Funktionen angewiesen ist, die die Hersteller ihren Geräten mitgegeben haben.
Das wurde auch von Softwarefirmen erkannt; sie wollen den Benutzern mit entsprechenden Programmen diese Aufgabe erleichtern oder sogar ganz abnehmen. Eines dieser Programme ist der „Print Shop" von Broderbund.

2. Arbeiten mit dem „Print Shop"

Ganz offensichtlich haben sich die Programmierer darüber Gedanken gemacht, was möglichst viele Druckerbesitzer wollten, wenn sie nur könnten. Herausgekommen ist ein Programmpaket, aus dessen Bestandteilen sich jeder etwas Passendes heraussuchen kann. Das Ganze läuft mit mit einer vorbildlichen Bildschirmführung, die die Anleitung vergessen läßt, die auch Fehlbedienungen sicher abfängt und nur mit einem freundlichen, aber bestimmten Dudelton quittiert.
Gesteuert wird das Programm durch die Pfeiltasten oder durch ein externes Eingabegerät wie den Joystick, ein Grafiktablett, z.B. Koala-Pad, oder die Maus.
Dank der stets angebotenen, unterschiedlichen Optionen findet man sich immer sicher im Programm zurecht. Das beginnt bereits bei der Installation. Aus einer Liste mit einer Vielzahl von Druckern und Interfaces wird die vorhandene Kombination per Knopfdruck ausgewählt. Ist diese kleine Pflichtaufgabe erfüllt, druckt Print Shop eine nette Begrüßungsbotschaft. Danach findet man sich im Hauptmenü wieder, wo die verschiedenen Programm-Module ausgesucht werden, die die Erfüllung der Druckwünsche versprechen.
Die Bedienung ist in allen Teilen möglichst einheitlich gestaltet, so daß man sich schnell im Programm auskennt. Eine bereits getroffene Wahl ist jederzeit durch die ESC-Taste wieder rückgängig zu machen, so daß gelegentliche Fehler nicht in kleine Katastrophen ausarten, die einen unnötigen Neubeginn bedingen.

3. Die Programm-Module

3.1. Greeting Cards

Für Einladungen, Grüße oder Glückwünsche wählt man *Greeting Cards*. Wie in jedem Block stehen auch hier alle grafischen Elemente (60 Bilder, Symbole und Muster) zur Verfügung, dazu kommen diverse Umrandungen aus Mustern, Linien und Streifen. Für Abwechslung und individuelle Kombinationen ist also ausreichend gesorgt, im übrigen sind inzwischen zwei Ergänzungsdisketten mit jeweils weiteren 120 Grafikelementen erschienen.
Ist die Karte gestaltet und mit dem gewünschten Text versehen (Schriftarten werden per Knopfdruck aus der Schriftenliste gewählt), wird sie nach einer kurzen „Denkpause" des Computers so geschickt geviertelt und gedreht auf das Papier gezaubert, daß nach zweimaligem Falten eine korrekt bedruckte Klappkarte entsteht, bei der Vorder- und Rückseite, Innen- und Außenseite richtig und wunschgemäß bedruckt sind.

3.2. Sign und Banner

Weiterer Mitteilungsdrang kann sich der beiden Module *Sign* und *Banner* bedienen. Während Sign unter Benutzung aller Print-Shop-Eigenschaften die Gestaltung von „Schildern" bis zur Größe einer kompletten Druckseite gestattet, die sich z.B. auch gut als Deckblätter für eigene Manuskripte eignen, bedruckt Banner in Längsrichtung ganze Papierbahnen mit Schrift und Grafikelementen. Je nach Text sollte vorsorglich der Papiervorrat begutachtet werden, denn was der Drucker mehr oder weniger lautstark produziert, kann bis zu einigen Metern lang werden. Interessant sind derartige Papierschlangen zur Dekoration oder in der Werbung, z.B. in Schaufenstern.

3.3. Letterhead

Wer grafische Spielereien mag, wird sich über *Letterhead* freuen, mit dem Briefköpfe unterschiedlichsten Charakters produziert werden können. Die Aufteilung kann (in Grenzen) individuell gestaltet werden, und die in der Größe angepaßten Grafikelemente wirken zum Teil recht putzig.

3.4. Screen Magic

Die letzten beiden Blöcke des Programmpakets können die eigene Kreativität unterstützen.
Screen Magic erzeugt unterschiedliche Kaleidoskop-Bilder, die dann im gewünschten Augenblick „eingefroren" und weiterverarbeitet werden können. Druckt man die Bilder dieses „Zauberbildschirms" aus, ist man zunächst wahrscheinlich enttäuscht, denn das Papierabbild fällt gegenüber der Monitoransicht stark ab. Erst das Hinzufügen von Schrift versöhnt wieder, denn jetzt steht diese deutlich dominierend vor dem phantasievollen, aber blassen Hintergrund.

3.5. Graphic Editor

Zu guter Letzt können inzwischen verwöhnte „alte Druckhasen" noch mit dem *Graphic Editor* eigene Grafiken gestalten oder vorhandene Bildelemente verändern. Die Möglichkeiten dieses kleinen Editors sind jedoch begrenzt und stellen eher eine Referenz an diejenigen Benutzer dar, die sich partout nicht mit den bereits vorhandenen Fix-und-Fertig-Grafiken zufrieden geben wollen. Immerhin sind damit eigene kleine Kreationen, wie z.B. das eigene Monogramm ohne weiteres machbar. Aus dem Graphic Editor heraus können Grafiken problemlos abgespeichert oder geladen werden.

Fazit

Insgesamt stellt der Print Shop eine außerordentlich komfortable Möglichkeit dar, Grafik selbst zu nutzen. Die Ergebnisse können sich durchaus sehen lassen. Wer das Programm nicht kennt und noch nicht weiß, mit welch geringem Aufwand seitens des Benutzers diese Ergebnisse erzielt werden, wird mit bewundernden Komplimenten nicht sparen.
Das Programm kostet DM 139,–, und das ist es auch wert.

The Print Shop Companion –

Die Print-Shop-Erweiterung

getestet von Hans-Martin Eng

1. Gegenstand

Muß ich Ihnen den Print Shop noch vorstellen? Ich glaube nicht; dieses von Broderbund vertriebene Programm dürfte inzwischen trotz einiger Unzulänglichkeiten einen recht hohen Bekanntheitsgrad erlangt haben. Diesen Unzulänglichkeiten rückt nun der Print Shop Companion zu Leibe.

1.1. Lieferumfang

Der Käufer des Print Shop Companion erhält einen Karton, in dem sich eine beidseitig beschriebene Diskette, ein Handbuch, zwei Diskettenaufkleber für Backup-Disketten und diverses Werbematerial befinden. Unerläßlich für die Benutzung des Programms ist der Besitz der original Print-Shop-Diskette.

1.2. Erstinstallation

Die Erstinstallation des Print Shop Companion nimmt nur wenig Zeit in Anspruch. Nach dem Booten der „Companion"-Diskette wird man aufgefordert, die „Print Shop"-Diskette einzulegen. Diese wird nun dahingehend modifiziert, daß Umrandungen und Zeichensätze von anderen Disketten in eigene Kunstwerke integriert werden können. Gleichzeitig wird die Hardwarekonfiguration, die auf der original Print-Shop-Diskette eingestellt war, auf die „Companion"-Diskette übertragen. (Käufer, die gleichzeitig Print Shop und Print Shop Companion gekauft haben, müssen also zuerst die Hardwarekonfiguration auf der Print-Shop-Diskette einstellen und erst anschließend den Print Shop Companion konfigurieren.)

2. Programme

Auf der Print-Shop-Companion-Diskette befinden sich folgende Programme:

2.1. Graphic Editor+

Die Kommandos des alten Grafikeditors sind reichlich kümmerlich. Außer Grafik laden/speichern, Punkt setzen/löschen und Grafik löschen konnte man eigentlich nichts tun. Der neue Grafikeditor wartet mit einer Fülle neuer Editiermöglichkeiten auf. Das beginnt beim Linienziehen und endet beim Löschen und Einfügen einzelner Grafikspalten bzw. -zeilen. Zusätzlich können Grafikfiles von anderen Programmen eingelesen werden; Ausschnitte davon können ausgewählt und als Print-Shop-Grafiken weiterbearbeitet werden. Wie beim alten Grafikeditor sind alle Optionen in einer Menüleiste aufgeführt. Besonders gut gefällt mir die Möglichkeit, die Grafik in allen vier Richtungen zu scrollen.

2.2. Border Editor

Beim alten Print Shop konnte man zwischen verschiedenen Umrandungen für Postkarten, Briefköpfe etc. auswählen. Diese waren jedoch fest vorgegeben und konnten nicht editiert werden. Beim Print Shop Companion kann sich der Hobby-Designer mit Hilfe des Border Editors individuelle Umrandungen einfallen lassen. Im Border Editor stehen sämtliche Editiermöglichkeiten zur Verfügung, die der Graphic Editor+ anbietet.

2.3. Font Editor

Ein großes Manko des Print Shop ist meines Erachtens das Fehlen von Kleinbuchstaben und Umlauten in den Zeichensätzen. Dem hilft der Font Editor des Print Shop Companion zumindest teilweise ab. Man kann sich damit einen eigenen Zeichensatz zusammenbasteln. Die Bedienung des Font Editors ist genauso komfortabel wie die der beiden anderen Editoren des Print Shop Companion. Leider ist es jedoch aufgrund des beschränkten Speicherplatzes auch weiterhin nicht möglich, *gleichzeitig* Groß- und Kleinschreibung zu verwenden. Dazu müßte wahrscheinlich eine generelle Revision des Print Shop erfolgen.

2.4. Tile Magic

Kennern des alten Print Shop ist die Menüoption Screen Magic bekannt. Mit ihr wurden kaleidoskopähnliche Bilder auf dem Bildschirm erzeugt. Sie konnten auf Wunsch direkt ausgedruckt werden, es war jedoch prinzipiell unmöglich, diese Bilder als Print-Shop-Grafiken weiterzubearbeiten.
Die Option Tile Magic erzeugt auf ähnliche Weise direkt Print-Shop-Grafiken. Man kann sie als Grafikfiles abspeichern, mit dem neuen Grafikeditor modifizieren und in eigene Druckerzeugnisse integrieren.

2.5. Creature Maker

Creature Maker ist eine Option für die verspielteren Print-Shop-Benutzer: Man kann sich seine eigene „Kreatur" zusammenstellen, indem man Kopf, Rumpf und Fußteil verschiedener bekannter „Wesen" zusammenstellt. Angeboten werden Frankensteins Monster, ein Affe, ein Hase, eine Maus, ein Elefant, ein Gespenst, ein Hobo, ein Outlaw, ein Clown und ein Gnom. Die fertige Kreatur kann dann als Grafikfile abgespeichert, mit dem Grafikeditor weiterbearbeitet und somit in eigene Kreationen eingebunden werden (z.B.: Ein Clown, dessen hasenähnliche Pfote zu allem Überfluß eine Karotte in der Hand hält und der auf den nicht gerade zierlichen Füßen eines Elefanten steht, lädt zum Kindergeburtstag ein.)

2.6. Calendar

Mit dieser Option des Print Shop Companion können Sie sich Ihre eigene Mischung aus Termin- und Wandkalender erstellen. Zuerst bestimmt man eine Print-Shop-Grafik, die in die Kopfzeile des Kalenderblattes gedruckt werden soll. Der Monatsname ist veränderbar (z.B.: January in Januar), die Jahreszahl kann frei zwischen 1753 und 9999 gewählt werden, dann muß man sich entscheiden, ob ein Monats- oder ein Wochenkalender erstellt werden soll. Für jeden Tag kann man einen kleinen Texteintrag vornehmen, anschließend wird das Ganze ausgedruckt.

2.7. Mitgelieferte Grafiken

Auf der Print-Shop-Companion-Diskette befinden sich neben den obengenannten Programmen auch Beispielgrafiken, neue Zeichensätze und neue Umrandungen für den Print Shop. Diese können über die Option „From other disk" in eigene „Drucksachen" eingebunden werden.

3. Bewertung

3.1. Bedienerfreundlichkeit

Wie auch beim Print Shop verläuft die Bedienung des Print Shop Companion weitgehend über Menüsteuerung. Als Eingabeeinheit werden Tastatur, Joystick, Koala Pad und Apple-Maus akzeptiert. Für mich persönlich blieb nur die Wahl zwischen Tastatur und Joystick. Da mein Joystick über nicht allzu stabile Potentiometer verfügt, war für mich die Bedienung über Tastatur vorteilhafter. Jeder Anwender sollte selbst herausfinden, mit welcher Eingabeeinheit er besser zurechtkommt.

3.2. Handbuch

Das englischsprachige Handbuch gibt dem Käufer eine gründliche Einführung in die Bedienung des Print Shop Companion. Alle Kommandos und ihre Auswirkungen werden erläutert, die Anwendungen werden anhand von Beispielen (mit Bild) verdeutlicht. Beigelegt ist außerdem eine „Apple Reference Card" auf der die einzelnen, auf der Print-Shop-Companion-Diskette abgespeicherten Zeichensätze und Umrandungen abgebildet sind.

Fazit

Zu einem Preis von ca. DM 120,- wird der Print Shop um einige nützliche Details erweitert – aber eben nur *erweitert*. Man benötigt unbedingt noch den original Print Shop, da man sonst die wunderschönen Grafiken nicht ausdrucken kann. Der schlägt dann nochmals mit ca. DM 140,- zu Buche. Betrachtet man aber die übrigen Softwarepreise, so liegt das Paar Print Shop und Print Shop Companion noch immer im unteren Preisbereich. Alles in allem ist der Kauf des Print Shop Companion durchaus empfehlenswert.

EXTRA K

Anwendungen für die erweiterte 80-Zeichenkarte

von Franz-Josef Hüskens

Viele Apple-IIe-Besitzer haben Ihren Computer mit einer erweiterten 80-Zeichenkarte auf 128K Speicher ausgebaut. Einige Anwendungsprogramme für den Apple benutzen diese Speichererweiterung automatisch, selbstgeschriebene Programme können jedoch nur mit Hilfsroutinen auf diesen zusätzlichen Speicher zugreifen. Hier setzt die Utility-Sammlung EXTRA K der amerikanischen Firma Beagle Brothers ein. Alle EXTRA-K-Programme sind sowohl unter DOS 3.3 als auch unter ProDOS lauffähig. Die mitgelieferte Diskette ist beidseitig bespielt und enthält auf beiden Seiten die Programme für das jeweilige Betriebssystem. Die folgenden, kurzgehaltenen Funktionsbeschreibungen enthalten auch Hinweise, unter welchem Betriebssystem die Programme lauffähig sind.

Aux.Mem.Check stellt unter DOS 3.3 fest, ob auf einem Apple IIe mit 128K (oder IIc) oder auf einem „normalen" Apple IIe gearbeitet wird. Die Ermittlung des Speicherausbaus unter ProDOS ist einfacher und wird im Handbuch erklärt.

Disconnect.RAM entfernt die von ProDOS automatisch geschaffene RAM-Disk, da alle Extra-K-Programme den Speicherplatz der Karte benötigen.

Disk.Compare dient zum Vergleich zweier Disketten. In zwei bis drei Diskettenzugriffen wird geklärt, ob eine DOS-3.3- bzw. ProDOS-Diskette den gleichen Inhalt hat wie eine andere Diskette des gleichen Betriebssystems. Das Resultat des Vergleichs wird auf dem Bildschirm im Track/Sector- bzw. Block-Format angezeigt (wahlweise hexadezimale oder dezimale Darstellung). Von diesem Bildschirm kann eine Hardcopy angefertigt werden.

Disk.Copy ist ein Disketten-Kopierprogramm, das Apple-Disketten (auch Pascal- und CP/M-Disketten) bei Verwendung von zwei Laufwerken in ca. 35 Sekunden kopiert.

Disk.Format formatiert ProDOS-Datendisketten. Im „Mini-Menü" kann man Steckplatz und Laufwerk auswählen, den Formatiervorgang starten oder das Programm beenden. Das Programm ist einfacher zu benutzen als der ProDOS-Formatter; es kann jederzeit von Applesoft aus aufgerufen werden.

Extra.Apple „verdoppelt" Ihren Rechner, indem es aus einem 128K-Apple zwei 64K-Apple macht. Dabei ist es möglich, daß „Apple 1" unter DOS 3.3 und „Apple 2" unter ProDOS arbeitet. Beide können auch unter dem gleichen Betriebssystem arbeiten. Da Extra.Apple ein „Umschalten" zwischen den beiden Rechnern gestattet, können Daten, Programme, Bilder usw. schnell und einfach von einem Betriebssystem zum anderen übertragen werden.

Transfer erlaubt das Kopieren von BASIC- oder Maschinenprogrammen vom „Apple 1" zum „Apple 2" (s.o.). Bei Binärprogrammen kann es u. U. zu Problemen kommen, für die im Handbuch jedoch Lösungen aufgezeigt werden.

Extra-Screens gestattet die schnelle Anzeige beliebiger Bildschirmdarstellungen. Es ist möglich, je nach Bildschirmformat 3 bis 62 verschiedene oder gleiche Bilder in den oberen 64K des Speichers abzulegen und diese dann bei Bedarf aufzurufen. Dazu erweitert Extra.Screens das Applesoft-BASIC um neun weitere Befehle, die durch den Ampersand-Vektor (&) angesprochen werden. Mit diesen Befehlen ist es möglich, Bilder in der Karte zu speichern oder aus der Karte zu laden, alle in der Karte gespeicherten Bilder an einem Stück auf Diskette zu speichern oder von Diskette zu laden, alle gespeicherten Bilder zu listen, Bilder umzubenennen, einzelne Bilder zu löschen, alle Bilder zu löschen und zu sehen, wieviel Platz in der Karte noch verfügbar ist. Damit man nicht unvorbereitet mit Extra.Screens arbeiten muß, enthält die Diskette eine Demonstration der Möglichkeiten des Programms in Text und Grafik.

Screens.Crop ist eine Routine, mit der Grafik- oder Text-Bilder aufgeteilt werden können. Man kann z.B. eine Hires-Grafik in viele kleine Einzelbilder zerlegen und diese dann speichern.

Extra.Variables vergrößert den verfügbaren Speicher für BASIC-Programme, indem die Programme im unteren 64K-Speicher, die zugehörigen Variablen jedoch in den oberen 64K untergebracht werden. Für die Programme stehen bei „normal" geladenem DOS 3.3 dann maximal 93K (statt 36K) zur Verfügung. Extra. Variables ändert nichts am Verhalten des Applesoft-Interpreters, nur die Ausführung der Befehle FRE, LOMEM: und HIMEM: wird geändert. Vorteile des Programms: Die von einem BASIC-Programm vereinbarten Variablen werden nicht gelöscht, wenn ein anderes Programm mit dem DOS-Befehl „Run NAME" gestartet wird. Die Benutzung des DOS-3.3-Programms bzw. des ProDOS-Befehls „Chain" ist nicht mehr erforderlich.

Peek.And.Poke und **Peek.And.Poke.X** erweitern Applesoft um die beiden Kommandos &Peek und &Poke. Damit kann in den oberen 64K nach Herzenslust gepeekt und gepokt werden. Durch die entsprechende Syntax der Kommandos (&Peek Adresse, Ausdruck, Bank) kann sogar die Language Card der oberen 64K angesprochen werden. Das erste Programm arbeitet mit normalem Applesoft, während „Peek.And.Poke.X" zu Extra.Variables kompatibel ist.

Extra.Store verändert das BASIC.System so, daß auch die Befehle Store und Restore bei Benutzung von Extra.Variables richtig arbeiten.

FP simuliert unter ProDOS den gleichnamigen DOS-3.3-Befehl.

Hybrid.Create „halbiert" Disketten und stellt die eine Hälfte unter ProDOS-, die andere unter DOS-3.3-Verwaltung.

Logbook protokolliert sämtliche Aktivitäten, die über den Bildschirmspeicher laufen. In den oberen 64K können dazu mehr als 47.000 Zeichen untergebracht werden. Das Protokoll kann jederzeit unterbrochen und wieder gestartet werden. Außerdem kann Logbook als Notizblock benutzt werden. Verschiedene Befehle gestatten das An- und Abschalten sowie das Anzeigen des Inhalts und das Löschen des Logbooks. Beim Listen des Protokolls kann die Information seiten- und zeilenweise auf- und abwärts gescrollt werden, man kann zum Anfang oder Ende des Logbuchs springen, bestimmte Zeichenfolgen können im Speicher gefunden und bis zu 10 Zeilen markiert werden.

Spooler ist eine Routine, die nach Angaben des Handbuchs die oberen 64K des Apple IIc zu einem Drucker-Zwischenspeicher umfunktioniert. Damit wird es mit Einschränkungen möglich, am Apple IIc zu arbeiten, während eine Druckroutine läuft.

Fazit

Mit Extra K ist den Beagle Brothers eine, für ca. DM 140,– recht preiswerte Utility-Sammlung gelungen, die es auch dem BASIC-Programmierer gestattet, den zusätzlichen Speicher der erweiterten 80-Zeichenkarte einfach und in vielfältiger Weise zu nutzen. Das 53seitige Handbuch ist in leicht verständlichem Englisch geschrieben und erklärt einfach und auch für Computerlaien verständlich, wie die Programme benutzt werden. Bei den meisten Routinen werden auch die programmspezifischen Fehlermeldungen erläutert.

peeker Börse

Gelegenheitsanzeigen / Kleinanzeigen

Sie können unter dieser Rubrik zu einem besonders günstigen Preis

- Ihre Hardware und Software verkaufen
- Ihre Hard- und Software suchen
- Kontakte knüpfen und vieles mehr

Musteranzeige privat (nicht gewerblich)

1 Druckzeile à 32 Buchstaben nur DM 5,50 zuzügl. ges. MwSt. Mindestens 2 Druckzeilen

Beispiel:

> Verkaufe neuwertigen Typenraddrucker mit Apple-Interface. Preis auf Anfrage. Tel. 007

nur DM 18,81 inkl. MwSt.

Musteranzeige gewerblich

1 Druckzeile à 32 Buchstaben nur DM 11,– zuzügl. ges. MwSt. Mindestens 2 Druckzeilen

Beispiel:

> Neu im Angebot: Professionelle, separate Tastatur für Apple II plus 16 Funktionstasten und separatem Ziffernblock. Fa. Keyboard & Co.

nur DM 62,70 inkl. MwSt.

peeker

Peeker-Börse

AUFTRAG FÜR KLEINANZEIGEN

Bitte veröffentlichen Sie in der nächsterreichbaren Ausgabe nachstehenden Text unter folgender Rubrik:

☐ suche Hardware ☐ suche Software ☐ Verschiedenes ☐ gewerblich
☐ biete Hardware ☐ biete Software ☐ Chiffre ☐ nicht gewerblich

Bitte den Text mit Schreibmaschine oder in Druckbuchstaben ausfüllen

Jeweils 32 Buchstaben pro Zeile – einschl. Satzzeichen und Wortzwischenräume. Bitte Absender nicht vergessen. Mindestens 2 Zeilen (1 Druckzeile à 32 Buchstaben DM 5,50 nicht gewerblich, DM 11,– gewerblich + MwSt.) – Chiffregebühr DM 6,– + MwSt.

peeker

Produkt-Karte

Zu der in **Peeker**, Heft _____, Seite _____ erschienenen

Anzeige über _____

bitte ich um detaillierte Information.

Ich wünsche ☐ Prospekt, Datenblatt ☐ Preisliste ☐ schriftliches Angebot ☐ tel. Rückruf

Menge	Produkt und Bestellnummer	à DM	gesamt DM

gebe ich nebenstehende Bestellung unter Anerkennung Ihrer in der Anzeige genannten Liefer- und Zahlungsbedingungen auf.

Unterschrift (für Jugendliche unter 18 Jahren der Erziehungsberechtigte)

peeker

Info-Karte

Ich interessiere mich für Beiträge über

☐ Apple ☐ IBM ☐ Atari ☐ _____

Peeker-Börse

Vorname, Name

Firma

Straße

Wohnort

PLZ/Ort

Bitte veröffentlichen Sie den umstehenden Text von _____ Zeilen à _____ DM in der nächsterreichbaren Ausgabe vom **Peeker**

Bei Angeboten: Ich bestätige, daß ich alle Rechte an den angebotenen Sachen besitze

Datum Unterschrift

Produkt-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

Anschrift der Firma angeben, bei der Sie bestellen bzw. von der Sie Informationen wünschen

Info-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

Bitte freimachen

POSTKARTE

Peeker-Börse
Anzeigen-Service

Dr. Alfred Hüthig Verlag

Postfach 10 28 69

6900 Heidelberg 1

Bitte freimachen

POSTKARTE

Inserent

Straße

PLZ/Ort

Bitte freimachen

POSTKARTE

Peeker
Redaktion

Dr. Alfred Hüthig Verlag

Postfach 10 28 69

6900 Heidelberg 1

Produkt-Karte

Wünschen Sie weitere Informationen zu einer der im Heft erschienenen Anzeigen?

Nichts einfacher als das. Produkt-Karte ausfüllen, frankieren und an den Inserenten (nicht an die Peeker-Redaktion senden.

Bitte aber nicht vergessen:
Kreuzen Sie an, welchen Informationswunsch Sie haben.

Damit erleichtern Sie dem Hersteller eine gezielte Beantwortung Ihrer Anfrage

Zum Schluß tragen Sie auf der Rückseite die genaue Anschrift des Inserenten und Ihren Absender ein.

PEEKER

mhe

und es geht doch!

MS-DOS für **APPLE II**

8086 Erweiterungskarte C 86

- 16-Bit Einplatinencomputer
- normale Slotkarte
- MS-DOS und CPIM 86
- 512 KB RAM-bestückt
- als RAM-Floppy verwendbar
- extrem schnell
- nur 998,– DM

Floppy-Controller FDC 97

- Controller mit APPLE- und Shugart Bus
- voll APPLE-kompatibel
- bearbeitet alle IBM-Formate
- bis zu 800 KB pro LW (2 x 80 T)
- div. Utilities im 16-KB EPROM
- nur 350,– DM

Einführungsangebot

bis zum 15. 5. 1987
Paketpreis: 998,– DM
C 86 : 698,– DM

Wir führen auch **IBM**-kompatible Produkte zu **Superpreisen** z. B. **NEC**-Festplatte 65 ms + Controller 1200,– DM

Angebot solange Vorrat reicht!

Marcus Herman Elektronik
Ellwanger Str. 12 · 1000 Berlin 46
Schnellversand:
Telefon (0 30) 774 74 84

Economy Software AG

Kaiserstr. 21, 7890 Waldshut, Tel. (0 77 51) 79 20

Frei-Programm- und Shareware-Zentrale
Über 30 000 Programme für IBM PC/Kompat, Apple II, Macintosh Atari ST, Commodore C64/C128, Amiga. Programme für Beruf, Geschäft, Heim und Schule.
Zum Kennenlernen guter Frei-Programme:

10 beliebte Programme für DM 10.–.

Dazu **gratis:**
Katalog über Frei-Programme (Public Domain) und professionelle Shareware auf Diskette(n) einschl. Sachgebiets-Verzeichnis im Wert von DM 10.–.
Dieses Kennenlern-Angebot erhalten Sie gegen Einsendung von DM 10.– (bar oder Scheck).

Bitte unbedingt Computermodell angeben.

APPLE & CP/M-80 & MS-DOS SOFTWARE & HARDWARE

z. B. für APPLE II und Kompatible
Wir liefern die RAM-Karte (AE) für den Apple IIe mit max. 3 MB (Appleworks mit mehr als 2 MB)! 64-K-Ausf. . . . . DM 650.–
SpeeDemon 3.56 MHz Coproz. für II+/e (McT) . . . . . DM 700.–
Anpassung für Appleworks 1.2 auf dem II+/e.
Original oder mit externer Tastatur. Anpassung in deutsch für SATURN 128 K und IBS AP33 1 MB! . . . . . DM 170.–
UPC-Programmer-Card 2716-128 komfortabel . . . . . DM 380.–
72 I/O Port Card programmierbar DOS+CP/M . . . . . DM 280.–
AD 16 Ch. 12 Bit, schnell! (Applied Eng.) . . . . . DM 1150.–
PKASO/U-Printer-Karte (IS) . . . . . DM 550.–
CP/M-Plus-Card, 6 MHz, 64 K, CP/M 3.0 (ALS) . . . . . DM 1150.–
Timemaster II H. O., **die** Uhrenkarte! (AE) . . . . . DM 540.–
ELF kompl. Statistik-Software (TWG) . . . . . DM 500.–
Prime-Plotter-Grafik-Software (Primesoft) . . . . . DM 900.–
Z-RAM 512 K für APPLE IIc (AE) . . . . . DM 1250.–

z. B. für IBM und Kompatible
APPLE Turnover (Vertex) Lesen/Schreiben von Apple Disks im IBM PC & Komp. . . . . DM 1000.–
XENO-COPY plus (Vertex) Lesen/Schreiben div. CP/M & MS-DOS Formate im IBM . . . . . DM 450.–
ELF PC kompl. Statistik Software (TWG) . . . . . DM 500.–
PROM Blaster 28-Pin (Apparat Inc.) . . . . . DM 620.–

z. B. für alle Systeme
Printerchanger 3 parall. Drucker auf 1 Micro inkl. Kabel/Netzteil (Keyzone) . . . . . DM 570.–
Printersharer 3 Micros auf 1 parall. Drucker inkl. Kabel/Netzteil (Keyzone) . . . . . DM 460.–
Shufflebuffer 64 K (IS) . . . . . DM 1250.–

Wir sind Import-Spezialisten und bieten Ihnen eine große Auswahl an Software und Hardware bedeutender Hersteller aus den USA und England.
Informationen gegen DM 3.– in Briefmarken.

WEISS COMPUTER **Dipl.-Psych. Karl-Heinz Weiß**
Am Wiesenhof 17, 2940 Wilhelmshaven, Tel. 0 44 21/8 31 79

Erster aaa „little big" Ableger
für Apple II/+/e, Basis 108 und compatible

Sicher. – Wir haben uns bei der Entwicklung u. Fertigstellung unserer **Mega Ramcard II**® etwas Zeit gelassen, damit Sie jetzt auch Zeit gewinnen. Und das hat sich gelohnt!
Die Tests, aber auch die mitgelieferten Ramdisktreiber-Programme für folgende Betriebssysteme, beweisen es:

- Apple DOS 3.3
- Diversi DOS 2-C/4-C
- Apple ProDOS 1.0.1, 1.1.1
- Apple Pascal 1.1
  Apple Pascal 1.2/64 K, 1.2/128 K
  Apple Pascal 1.3/64 K, 1.3/128 K
- CP/M 2.2 für
  Microsoft Softcard, CP/M 2.20, 2.23
  Microsoft premium card, CP/M 2.26
  Microsoft Softcard II, CP/M 2.28

Sogar nach einem „Systemabsturz" oder Betriebssystemwechsel bleiben alle Dateien erhalten!
Peripherieslot und Speicherkapazität werden automatisch erkannt, sodaß das lästige Eingeben entfällt.
Mega Ramcard II® ist mit den erphi-AFDC2/AFDC3-Controller vollständig kompatibel, u.v.m.

Händleranfragen schriftlich erwünscht.

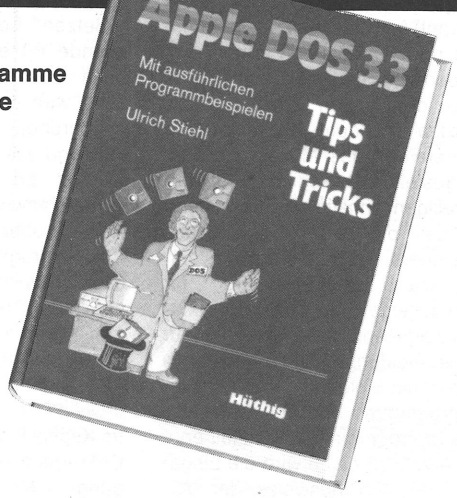
Mega Ramcard II, 1 MByte bestückt + Software (s.o.) + deutsches Handbuch, II/+/e	**878,–**
Mega Ramcard II, 256 K-RAM best. (bis 1 MB aufr.) + Software (s.o.) + dt. Hb., II/+/e	**498,–**
Apple-Works Anpassung	**69,–**
Disk II-Siemens-Laufwerk im Geh.+ Kabel f. orig. + comp. Contr. geeignet, II/+/e	**298,–**
Disk II-Siemens-Laufw.+ Contr.+ Kabel, II/+/e	**359,–**
Erphi-AFDC2-Controller + Autopatch Software + deutsches Handbuch, II/+/e	**198,–**
Disk II-Philips Laufwerk o. Geh., 2 x 80 Track-640 KB f. AFDC2 o. 3 modifiziert, II/+/e	**298,–**
Erphi-DuoDisk 1,2 MByte im Gehäuse + Erphi-AFDC3-Controller, II/+/e	**898,–**

REPARATUREN an Apple + Compatiblen Geräten + Zubehör führt unser Spezialteam garantiert zuverlässig + besonders kostengünstig aus. **Sprechen Sie mit uns. Kostenvoranschlag auf Wunsch!**

aaa electronic gmbh
Telex 07 72 642 aaa-d
Habsburgerstraße 134
7800 FREIBURG, Tel. (07 61) 27 68 64
Bauelemente – Bausätze – µP's
Meßgeräte – Zubehör – Fachliteratur
Fachgeschäft für Elektronik + Mikrocomputer

Hüthig BUCH-TIP

Wegen der neuen Programme paßt die neue Begleitdiskette nur zur 3. Auflage.



Apple DOS 3.3 — Tips und Tricks

von U. Stiehl
3., völlig überarb. Aufl. 1986, X, 203, mit zahlreichen, ausführlich kommentierten Programmlistings, kart., DM 28,–
Begleitdiskette ebenfalls DM 28,–

Dr. Alfred Hüthig Verlag · Postf. 10 28 69 · 6900 Heidelberg 1

aaa electronic gmbh
Neue Preise #3

LEER/BESTÜCKT

Industrie-Standard IBM-PC/XT+AT-compatible COMPUTER+Zubehör in Industrie-Qualität (X=XT, A=AT, X/A=XT+AT)
Mainboard XT-256, 8088 CPU, OK (bis 256K-on board, oder 640 K mit Addr. auf Karte aufr.) + BIOS + 8 Slots, geprüft 79,–/227,–
Mainboard XT-256 mit 256 K-RAM, sonst. w.o., gepr. 317,–
Mainboard XT-640, 8088 CPU, OK (bis 640 K-on board) gepr. 89,–/237,–
Mainboard XT-640 w.o. jedoch 256 K-RAM best. + gepr. 297,–
Mainboard XT-640 w.o. jedoch 640 K – Mainboard, geprüft 407,–
Mainboard TURBO XT-640 -4,7/8 MHz. OK, best. + gepr. 99,–/276,–
Mainboard AT-1MB, 6/8 MHz, 80286 CPU, 16/24 bit (mit Socket f. 80287) 512K-RAM best (bis1MB aufrüstb.)+BIOS+Uhr zu Akku+8 Slots,gepr.**1229,–**
64KB-RAM-Chip-Aufrüstsatz (9 Stück) einzeln geprüft! **X/A** 32,–
256 KB-RAM-Chip-Aufrüstung (9 Stück) einzeln geprüft! **X/A** 69,–
Disk-Controller gepr. (f. 2x360 KB-Drives), **X** 77,–
Disk-Controller gepr. (f. 4x360 KB-Drives)(Extra Kabel=39,–), **X** 34,–/94,–
Disk-Controller gepr. (f. 2x1,2 MB- o. 360 KB-Drives), **X/A** 229,–
Disk-Controller gepr. (f. 2x1,2 MB- o. 360 KB-Drives), **A** 39,–/169,–
Harddisk-Controller f. 2 HD, **X** 287,–
Harddisk+Floppy-Controller (f HD+2DD) gepr., **A** 597,–
Monochrome Karte geprüft, **X/A** 49,–/149,–
Monochrome Graph.+par.Print. Karte (Hercules comp.), **X/A** 55,–/168,–
Color RGB + Video Graphikkarte + Lightpen-Ausg. gepr., **X/A** 55,–/128,–
Parallel Printer-Karte gepr. o. Kabel (Extra Kabel = 49,–), **X/A** 29,–/ 49,–
2xRS 232 Interface (1 davon bestückt), **X/A** 29,–/ 73,–
2xRS 232 (1 davon bestückt) + par. Printer Karte gepr., **X/A** 138,–
512K-RAM-Karte gepr. (0KB)(mit 64 KB-RAM bestückbar), **X** 55,–/ 88,–
512K-RAM-Karte gepr. (0KB)(mit 256 KB-RAM bestückbar-kurz), **X** 114,–
3 MB-RAM-Erweiterungsk. gepr. (OK), (m. 256 K-RAM bestückbar), **A** 317,–
2,5 MB-Multifunktionskarte, OK, +RS 232, +par. Printer, gepr., **A** 397,–
384K-Multifunktionskarte, RS 232, par. Printer, Interf. (Game/Joystick Port, Real Time Clock gepuff., OK+RAM-Disk/Spooler/Uhr Softw., **X** 55,–/189,–
Disk Multi I/O Karte f. 2 Disk-Drive, 2xRS 232, 1 par. Printer Int., Game/Joystick Port, Real Time Clock/Kalender gepuff.), **X** 59,–/169,–
Clock Karte gepuffert, geprüft, **X** 69,–
I/O Plus-II-Karte, 2xRS 232, (1 x best.), par. Printer-Interf., Game /Joystick Port, Real Time Clock gepuffert, geprüft, **X** 179,–
I/O Plus-Karte, 2xRS 232 (1xbest.), par. Printer-Interf., Game/Joystick Port, geprüft, **A** 159,–
EGA-comp. Enhanced Graphic Adapter Karte, mit 256 K-RAM-Bildschirmspeicher, Monochrome-Color Graphik (640x350 Punkte – 16 Farben) + par. Printer Int., voll best. + gepr., **X/A** 69,–/495,–
AD/DA-Wandler, 12 bit, 16/1, **X/A** 295,–
AD/DA-Wandler, 14 bit, 16/2, **X/A** 475,–
Option Board compatible Karte incl. Disk+Manual, (zum sichern kopiergeschützter Disketten, Kopierschutz wird mitübertragen), **X/A** 347,–
IEEE-Interface+Manual (Extra Kabel=89,–), **X/A** 798,–
Eprom Writer, (f. 1 St. 2716/32/64/128), **X/A** 275,–
Eprom Writer, (f. 4 St. 2716/32/64/128/256-512 gleichzeitig), **X/A** 429,–
Prototype/Lochraster Leerkarte, durchkontaktiert, lang, **X/A** 64,–
Prototype/Lochraster Leerkarte, durchkontaktiert, lang, **A** 84,–
Profi-Funktionstastatur ASCII + 15er Block ab 195,–– /dt. ab 225,–
135W/150W-Netzteil mit eingeb. Ventilator, kurzschlußf., **X** 195,–/215,–
200W-Netzteil mit eing. Ventilator, kurzschlußfest, **A** 375,–
360KB-Diskdrive Slimline(Track/Access Time <40ms),R/Wgepr., **X/A** 277,–
720KB-Diskdrive Slimline, 2x80 Tr. (TAT <2mS), R/W gepr., **X/A** 337,–
1,2MB-Diskdrive Slimline, 2x80 Tr. (TAT <2mS), R/W gepr., **X/A** 347,–
21 MB-NEC-Harddisk, X/A 979,–– **Micro Science, X/A** 849,–
21 MB-NEC-HD+HD-Contr., X 1.285,–– **MS+HD-Contr., X** 1.155,–
44 MB-NEC-Hardsk, X/A 1.957,–– **44 MB-NEC+DTC, X** 2.349,–
21 MB, Easy Card, Harddisk+Contr. in einem, anschlußfertig, **X/A** 1.298,–
Joystick f. IBM + Compatible, X/A 58,–
Maus incl. Testsoftware (MOUSE Systems comp.), **X/A** 158,–
Maus (Microsoft u. MOUSE Systems comp.), **X/A** 249,–
Met.-Gehäuse aufkl. (Ft f. 4-Erib.-Drives), solide Ausf., **X**=198,–/ **A**=298,–
+++PC/XT+AT = KOMPLETT-SYSTEME AUFGEBAUT UND GEPRÜFT+++
XT-640, 8088 CPU (incl. **Boot-Eprom) 256K-Ram incl. Lautsprecher, im IBM-look-like Met.-Gehäuse+Profi-Fkt. Tast.+135 W-Netzt. gepr. 777,–**
XT-640, w.o. +Contr. +360KB Drive+Color RGB+Vid. Graphik-Karte **987,–**
XT-640, w.o. jedoch statt Color mit Monochrome Karte 997,–
XT-640, w.o. jedoch mit Monochrome-Graphik-Printer-Karte 1.017,–
XT-640, 256 K-RAM+360 KB Drive+Disk Multi I/O+Color-Gr.-Karte 1.097,–
XT-640, 256 K-RAM+360 KB Drive+Color RGB+Video-Graphik-Karte+21 MB-Festplatte+Controller **2.097,–**
AT-1MB, 6/8 MHz, 80286 CPU, 512 K-RAM, im IBM-look-like Met.-Gehäuse+Profi-Fkt. ASCII-Tast+Netzt.+Color Graph.-Karte+1,2MB-Drive, gepr. **2.295,–**
AT-1MB, w.o.+21MB-Harddisk gepr. 3.785,–
Aufpreis für Turbo XT-640 -4,7/8 MHZ umschaltbares Board, **X** 39,–
Aufpreis für XT-640 M-board mit 640 K von 256 auf 640 K-RAM, **X** 119,–
Aufpreis f. EGA anstatt Color Graph.-Karte, **X/A** 475,–
Aufpreis f. NEC-360 K-Drive anstatt HS, **X/A** 75,–
Aufpreis f. NEC-1,2 MB-Drive anstatt HS, **X/A** 95,–
MS-DOS 3.1 Diskette incl. Manual, **X/A** 159,–
MS-DOS+GW-Basic 3.2 sowie 4 dt. org. Microsoft Hb. (Aufpreis in Verbindung mit Systemkauf), **X/A** 189,–
8087-Arithmetik-Prozessor, 5MHz, X=327,–/ 8028?-8MHz, A=869,–/5MHz, A 869,–
V 20-Prozessor, 8 MHz, X 37,–

APPLE-BUS COMPUTERPLATINEN + PERIPHERIE	LEER/BESTÜCKT
M-board II 48K, gesockelt, vollbest. + gepr.	57,–/237,––
M-board II 64K, gesockelt, vollbest. + gepr.	88,–/375,––
M-board II 64K + Z80 CPU, gesockelt, vollbest. + gepr.	95,–/378,––
M-board IIe 64K vollbestückt + gepr.	98,–/389,––
Superstarkes Netzteil – kurzschlußfest 5A=115,–– b.z.w. 7A=	145,––
Entwicklungslochraster-Leerplatine durchkontaktiert	34,––
16K-Speicher-Karte geprüft	18,95/ 59,––
Z-80 CPU-Karte geprüft (CP/M Vol. I.+II Manual dafür = 69,–)	18,95/ 74,––
Controller DOS 3.3-Karte geprüft	18,95/ 69,––
Auto-Controller DOS 3.2/3.3-Karte gepr.	24,–/ 97,––
Erphi AFDC 2 Controller + Autopatch Softw. + Handbuch, gepr.	198,––
Parallel-Drucker-Karte geprüft (Extra Kabel = 45,–)	18,95/ 74,––
Par.-Drucker Karte-Grappler comp. gepr. (Extra Kabel=45,–)	27,–/169,––
8bit par. Int. + 64K-Buffer + Kabel gepr. (Grappler comp.)	39,–/299,––
Seriell-Interface-Karte V24 geprüft	18,95/ 97,––
Super-Seriell-Interface Karte gepr.	34,–/178,––
80-Zeichen/24-Zeilen-Karte geprüft	18,95/129,––
80Z/24Z-Karte + Softswitch-Schalter, gestochen scharf, gepr.	24,–/138,––
80Z/24Z-Softwitch Karte gepr. für Ile	15,–/ 39,––
80Z/24Z + 64K-Ram+Softsw.-Karte best.+gepr. f. Ile	18,95/ 94,––
IEEE-488 Int. Karte gepr. (Extra Kabel=45,–)	29,–/297,––
PAL-Modulator/Color-Karte+UHF Modulator, geprüft	18,95/ 99,––
RGB-Color Karte gepr., (nur f. II+)	18,95/137,––
128K-Speicher-Leerkarte + Software + Manual	99,––
128K-Speicher-Karte + Software + Manual, geprüft	29,–/255,––
Mega Ramcard II, 256 KB best.+Software+dt. Hb., (bis 1 MB aufr.)	498,––
Mega Ramcard II, 1 MB best. + Software+dt. Hb., gepr.	878,––
Apple-Works Anpassung f. Mega Ramcard	69,––
EPROM-Burner (2716/32/64) geprüft	24,–/119,––
6522-VIA Karte geprüft	24,–/147,––
Clock Karte+Software+Manual, geprüft	24,–/ 99,––
Speech Karte+Software+Manual, geprüft	18,95/ 53,––
Musik Karte+Software+Manual	159,––
Wild Karte+Software+Manual (nicht f. Ile), geprüft	18,95/ 59,––
AD/DA-8Bit Karte+Software+ Manual, gepr.	29,–/269,––
Joystick f. II/+/e	69,––
UHF/TV-Modulator universell	37,50
Lüfter anclipsbar (220 V)	95,––
APOLLO//(48K) vollbest. + Gr/Kl vollbest. + gepr.	617,––
APOLLO//(48K) + Disk II F + Contr. +12"-Monitor	1.269,––
APOLLO//A (48K) + Gr/Kl + 15er-Tastatur vollbest. + gepr.	677,––
APOLLO//ASKF (48K) vollbest. + gepr. sep.Keyboard + Gr/Kl + 15er-Block mit Funkt.-Tasten im „IBM-look like" Gehäuse	835,––
Aufpreis f.o. Systeme von 48K zu 64K	55,––
Aufpreis von 48K zu 64K+Z80 CPU	89,––
Apollo Ile (64K) vollbest. und geprüft	675,––
Apollo Ile (64K)+80Z/24Z-Karte gepr.	698,––
Apollo Ile (128K) + 80Z/24Z-Karte geprüft	755,––
Apple Ile (64K)+15er-Tastatur vollbestückt und geprüft	735,––
Apollo Ile ASKF (64 K) vollbestückt und geprüft	ab 845,––
Apollo Ile (64K)+Disk II F+Contr.+12" Monitor	1.387,––
APPLE Ile 128K+80Z/24Z Karte+Softw.	1.598,––
Disk-Drives voll APPLE II, 100%ig kompatibel+R/W geprüft+Track Ø Sensor:	
Disk/+Contr. + Kabel DOS 3.3 (Siemens) im Geh.	359,––
Disk/(Siemens) im Geh. + Kabel	298,––
Disk//F-HS (High Speed 1/2 Höhe) im Gehäuse+Contr. +Kabel	379,––
Disk//F-HS (High Speed 1/2 Höhe) im Geh.+Kabel (Track Ac.<2mS)	318,––
Disk//F-HS-80 Track im Gehäuse + Kabel f. orig. Contr. geeignet	389,––
Disk//F-HS-80 w.o. + Contr. + DOS 3.3 + CP/M Patch-Softw.	489,––
Disk II-2x80 Track 640KB f. Erphi Controller modifiziert, o. Gehäuse	299,––
Erphi DuoDisk 1,2 MByte im Gehäuse + Erphi-AFDC 3-Contr., gepr.	898,––
APPLESOFT + Tutorial + Reference Manual engl.	75,––
APPLE-DOS 3.3 Manual engl.	25,––
APPLE-Pascal Reference + Operating Manual engl.	49,––
APPLE-Fortran Manual engl.	69,––
CP/M-Softcard Vol. I + II, Manual engl.	69,––
EPSON FX 800, 8bit/parallel 240 Z/S. NLQ + IBM comp.	1.398,––
EPSON FX 105+, 8bit/parallel 15" sonst w.o.	1.397,––
EPSON LQ-800, 24 Nadel, 180 Z/Sec. NLQ+IBM comp.	1.488,––
EPSON LQ-1000, wie 800 breite Ausf.	1.937,––
EPSON EX 800,	1.494,––
EPSON Traktoraufsatz für FX 80 + FX 85	129,50
APPLE/EPSON-Drucker Graphic-Interface + Kabel	169,––
MX 80/82, FX/RX 80, FX 85/800 Spezialfarbband-Kassette	15,95
NEC-Multisync 14" EGA, Farbmonitor	1.998,––

Wir führen verschiedene Monitore von Zenith, Phillips, NEC, COGA und Sanyo mit Video – TTL (IBM Komp.) oder Colormgang. Sowie umfangreiche Computer-Literatur (über 500 versch. Titel) – Bitte Preisliste anfordern! –

Disketten in Box + Aufkleber f. Wahl, 10er Pack/100er Pack/ Stückpreis:

5¼"	BASF	3M	FUJI	MAGNETICS	NEUTRAL
1X, SS/SD	2,79/ 2,59				
1D, SS/DD	2,89/ 2,69	2,88/ 2,78	4,27/ 3,97	–,980/–,936	–,885/–,835
2D, DS/DD	3,49/ 3,29	3,88/ 3,68	5,87/ 5,47	1,180/–,996	1,085/–,895
1D/96TPI	3,89/ 3,59	4,88/ 4,68			
2D/96TPI	4,39/ 4,19	4,68/ 4,38	7,87/ 7,67		
2HD/1,2MB	7,59/ 7,29	7,98/ 7,58	9,87/ 9,57		
1X, Hard	4,29/ 4,19				
8"					
1X, SS/SD	4,49/ 4,39	6,68/ 6,48	6,97/ 6,67		
1D, SS/DD	4,79/ 4,59	8,58/ 8,28			
2D, DS/DD	6,49/ 5,99	9,88/ 9,58	10,17/ 9,77		
1X, Hard	**4,49/ 4,39**	**6,68/ 6,48**			
3½"					
Einseitig	4,69/ 4,49	5,88/ 5,58	7,67/ 7,17		
Zweiseitig	6,19/ 5,89	7,28/ 6,88	9,17/ 8,57		

Das sind Preise!

5¼" Disketten-Archivbox für 10 Disks (f St./10 St.)	4,75/43,50
5¼" Disk-Kartelkasten Kunststoff (ca. 80 Disk)	26,50
5¼" Disk-K.K.Rauchglas, abschließb. (f. ca. 80-100 Disks)	39,50
2000 Bl. Tabpapier (24 cm x 12" eint.) weiß oder grün/weiß perf.	45,––
4000 Et.-Aufkl. doppelr. (107 x 36 mm auf 240 x 12" perf. Trägerp.)	66,50
4000 Etik.-Aufkl. einreih. (107 x 36 mm auf 125 x 12" perf. Träg.)	65,––

OSZILLOSKOPE HAMEG ab Lager

Bei Vorauszahlung frei Empfangsstation unversichert in der BRD, ausgenommen Papier und Etiketten, sonst N.N. + V.S. ab DM 30,––
Öffnungszeiten: Mo, Di, Do, Fr v. 10-18 h, Mi u. Sa v. 10-14 h. Ia Sa v. 10-18 h.
Telef. Best.: Mo, Di, Do, Fr von 10-19 h, Mi u. Sa wie Öffnungszeiten.
GEWÄHRLEISTUNG: 6 Monate, auf alle bei uns gekauften Geräte, durch unsere eigene Service-Werkstatt.

REPARATUREN an Apple + compatiblen Geräten + Zubehör führt unser Spezialistenteam garantiert zuverlässig + besonder kostengünstig aus
Sprechen sie mit uns. Kostenvoranschlag auf Wunsch!

aaa electronic gmbh
Telex 07 72 642 aaa-d
Habsburgerstraße 134
7800 FREIBURG, Tel (07 61) 27 68 64
Bauelemente – Bausätze – µP's
Meßgerate – Zubehor – Fachliteratur
Fachgeschäft fur Elektronik + Mikrocomputer

Visible Computer – ein Assembler zum Zuschauen

von Dagmar Berberich

1. Gegenstand

Visible Computer ist ein Maschinensprache-Lehrsystem, das dem systematischen Erlernen der Apple-II-Maschinensprache dient. Es besteht aus einem 149seitigen, spiralgehefteten Buch und einer Begleitdiskette und kostet zusammen DM 58,–.
Das Buch führt in 16 Kapiteln in die Programmierung des 6502-Prozessors ein. Die Begleitdiskette enthält ein Programm, das einen stark verlangsamten 6502-Prozessor simuliert; der Anfänger kann als „Zuschauer" die einzelnen Arbeitsschritte des Prozessors mitverfolgen und sich dadurch den Ablauf der Assemblerprogrammierung verdeutlichen.
Für das Simulationsprogramm wird ein Apple IIe/c/+ mit einem Diskettenlaufwerk benötigt; ein Drukker ist nicht unbedingt erforderlich.

2. Aufbau des Buches

Das Buch ist in 16 Kapitel und die Anhangskapitel A-F gegliedert.
Die ersten drei Kapitel sind als allgemeine Einführung in Bau und Funktionsweise eines Computers gedacht und können auch übergangen werden.
Die Kapitel 4 bis 16 bilden den Kern des Programmierkurses. Darin werden 6502-Maschinenbefehle mit steigendem Schwierigkeitsgrad behandelt; auf der Begleitdiskette werden die Befehle in einzelnen Simulationsprogrammen verdeutlicht. Am Ende des Kapitels 14 sind auf diese Weise fast alle der 56 Befehle des 6502 erläutert und demonstriert worden.
Kapitel 15 zeigt den Entwicklungsgang eines vollständigen, kleinen Programms von der Idee über Design- und Codierphase bis zum Assemblieren und Austesten.
Das Abschlußkapitel 16 enthält eine Liste gebräuchlicher Assembler, Tips zur Fehlersuche und Techniken, um BASIC- und Maschinenprogramme miteinander zu verbinden.

3. Inhaltsübersicht

Im ersten Kapitel wird erläutert, wann und wozu man Maschinensprache zweckmäßigerweise benutzt und wie man Arbeitsteilung zwischen BASIC und Maschinensprache betreiben kann.
Das nächste Kapitel befaßt sich mit den verschiedenen Zahlensystemen (dezimal, binär, hexadezimal) und den entsprechenden Umrechnungsmethoden. Im Kapitel 3 werden der Grundaufbau eines Computersystems und die Speicherliste des Apple II kurz erklärt. Der Autor veranschaulicht die Fachbegriffe anhand allgemein bekannter Beispiele (Karteikartenschrank usw.) und bedient sich einer lockeren Sprache, die im ganzen Buch beibehalten wird.

Kapitel 4 führt in die Arbeit mit dem 6502-Simulatorprogramm VISIBLE COMPUTER ein. VISIBLE COMPUTER (VC) besteht aus zwei Hauptteilen: der Simulator (ohne Prompt) führt die 6502-Programme aus, und der Monitor (Prompt „#") steuert den Simulator.
Nach dem Laden des Simulators werden auf dem Bildschirm die „Innereien" des Mikroprozessors dargestellt: 10 Register und ihre jeweiligen Inhalte als Hexadezimalzahl, die beiden Register des Speichers (16-Bit-Register für Adressen, 8-Bit-Register für Daten), eine Statuszeile, die aktuelle Kommandozeile, eine Zeile für Fehlermeldungen und 2 Fenster, in denen die Meldungen des Simulatorprogramms und der disassemblierte Programmtext erscheinen.
Im Anschluß daran wird die Eingabe und Wirkungsweise der VC-Kommandos beschrieben.

Im nächsten Kapitel wird gezeigt, wie die Speicheraufteilung des Apple aussieht, während ein VC-Programm läuft. Danach wird das Prüfen und Ändern von Speicherinhalten dargestellt. Mit „WINDOW MEM" wird ein Fenster zum Speicher geöffnet, das den Blick auf 16 Speicherstellen frei gibt.
Die schnellste Art, den Inhalt bestimmter Speicherplätze zu beschreiben oder zu ändern, ist das direkte Laden (Syntax: „Adresse Wert", in der angewählten hexadezimalen oder dezimalen Zahlenbasis, z.B. „806 4F"). Sollen mehrere Speicherstellen beschrieben werden, gibt man mit der EDIT-Funktion fortlaufend die Werte aufeinanderfolgender Speicherstellen ein.

Die Register des 6502 werden kurz in Kapitel 6 angesprochen, danach wird der Umgang mit dem Simulator anhand des ersten Beispielprogramms erläutert. Die nächsten Programme demonstrieren Lade-, Speicher- und Transferbefehle des Akkus und des X- und Y-Registers.

Kapitel 7 beschäftigt sich mit dem Prozessor-Status-Register (P-Register), dessen 8 Bits (Flags) eigene Namen haben und einzeln gesetzt werden. Implizite Befehle (1-Byte-Befehle) löschen oder setzen die C-, D- und I-Flags des P-Registers bzw. löschen das V-Flag. („Setzen" bedeutet, das entsprechende Bit erhält den Wert „1", beim „Löschen" oder „Zurücksetzen" wird dem Bit der Wert „0" zugeordnet.) Das Zero- oder Z-Flag und das Negativ- oder N-Flag werden bei jeder Lade- oder Transferanweisung konditioniert (gesetzt oder gelöscht).
Beispielprogramm 5 verdeutlicht das Disassemblieren. Ein disassemblierter Befehl besteht aus einem Mnemonic, dem Kurzwort für einen Maschinenbefehl, und dem Operanden.

Im Kapitel 8 geht es um Inkrement/Dekrement-Befehle und Verzweigungen. Mit DEX, DEY, INX, INY wird der Inhalt des X- bzw. Y-Registers dekrementiert bzw. inkrementiert, also um eins vermindert bzw. erhöht. Ein Anwendungsbereich dieser Befehle sind Zählschleifen und Wiederholungen. Beim 6502 gibt es insgesamt 8 Verzweigungsbefehle, je 2 für die C-, N-, Z- und V-Flags des P-Registers. Diese Befehle benutzen die relative Adressierungsart. Eine Verzweigungsanweisung besteht aus 2 Bytes: dem Opcode-Byte (das dem 6502 sagt, welches Bit er auf welchen Zustand testen soll) und dem Offset-Byte. Die Summe aus Offset-Byte und dem Wert des Programmzählers ergibt die Sprungadresse, an die verzweigt wird, wenn der Test positiv ausfällt. Sprünge können vorwärts und rückwärts erfolgen.
Die Geschwindigkeit von VC kann über die „STEP"-Anweisung reguliert werden.

Kapitel 9 befaßt sich mit den Adressierungsarten des 6502. Sie bestimmen, *wo und wie* die Daten zu finden sind, die ein Befehl benutzt.
Die Instruktionen JMP (Jump), JSR (Jump to Subroutine) und RTS (Return from Subroutine), um die es in Kapitel 10 geht, ermöglichen Sprünge im Hauptprogramm oder in ein Unterprogramm. JMP ist ein 3-Byte-Befehl, der die angegebene Adresse in den Programmzähler (PC) lädt und so zu jeder beliebigen Stelle im Speicher führen kann. Die Kombination JSR/RTS entspricht dem GOSUB/RETURN in BASIC, d.h. Sprung in ein Unterprogramm und – nach dessen Ablauf – Rückkehr zum nächsten Befehl des Hauptprogramms. Die Rückkehradresse wird zuvor auf den Stack geschrieben.
Der Stack (S-Register), der als Zwischenspeicher für 256 Bytes dienen kann, wird sehr bildlich und einleuchtend erklärt. Ebenso die Push- und Pull-Instruktionen, die das Statusregister lesen und verändern. JMP kann mit direkter (absoluter) und indirekter Adressierung benutzt werden. Bei indirekter Adressierung wird im Adreßteil des JMP-Befehls die Adresse genannt, an der die eigentliche Adresse des Operanden zu finden ist.
Kapitel 11 bespricht die Befehle für Additions- und Subtraktionsschritte (ADC = Add with Carry, SBC = Subtract with Borrow), die Befehle CLC (Clear Carry), SEC (Set Carry) und den Vergleichsbefehl CMP.
Die SHIFT- und ROTATE-Befehle ASL, LSR, ROL und ROR, die einzelne Bits statt ganzer Register verschieben, sind Thema von Kapitel 12. Hier werden auch die logi-

schen Verknüpfungen AND, ORA, EOR behandelt.

Indexierte Adressierungen, die im Kapitel 13 beschrieben werden, sind besonders bei der Benutzung von Arrays nützlich. Nehmen wir das Beispiel „LDA $0A00,X": Der Inhalt des X-Registers wird zur Basisadresse $0A00 hinzuaddiert. Diese Zahl gibt die neue Adresse an, deren Inhalt in den Akku geladen wird. Indexierungsbefehle können auch Basisadressen auf der Zero-Page ansprechen; durch den Fortfall des höchstwertigen Adreßbytes (MSB = 00) belegen diese Befehle nur 2 statt 3 Bytes. Die indirekte indexierte Adressierung und die indexierte indirekte Adressierung erweitern das Angebot an Adressierungsmöglichkeiten.

Im 14. Kapitel werden verschiedene Befehle angesprochen, die bisher noch unberücksichtigt blieben: NOP (No Operation) für Warteschleifen und als Füllstoff, die Interrupt-Befehle Reset, IRQ (Interrupt-Request) und NMI (non-maskable Interrupt) und der Break-Befehl BRK (Software-Interrupt).

Für die Berechnung von negativen Zahlen wird die Zweierkomplement-Schreibweise eingeführt.

Im vorletzten Kapitel wird die Entwicklung eines vollständigen, kleinen Maschinenprogramms am Beispiel eines Programms zur Tonerzeugung und des Sortierverfahrens „Bubble Sort" schrittweise erarbeitet.

Das letzte Kapitel nennt einige Assembler und geht auf den Apple-Monitor und das Maschinenprogramm Applesoft ein.

Anhang A zeigt die Speicherbelegung durch VC, Anhang B und C den ASCII-Zeichensatz und die VC-Monitor-Befehle. Der Simulator des VC wird in Anhang D erläutert. Die Fehlermeldungen des Systems werden in Anhang E aufgelistet. Anhang F gibt eine Einführung in das Assemblersystem ASSYST, das auf der Rückseite der Begleitdiskette mitgeliefert wird.

Leider fehlt dem Buch ein allgemeines Register, nicht einmal eine Übersicht über die behandelten Assemblerbefehle ist vorhanden. Will man einen bestimmten Befehl nachlesen, muß man das Buch mühsam durchsuchen. Gerade weil sich Visible Computer als Anleitung für Assembler-Anfänger versteht, dürfte es auf einen solchen Nachschlageteil nicht verzichten.

4. Das Programm

Das Simulatorprogramm führt auf dem Bildschirm die Instruktionen des 6502-Mikroprozessors einmillionmal langsamer aus als der Original-Prozessor. Der Assembler-Neuling kann deshalb schrittweise auf der Registeranzeige verfolgen, wie sich die einzelnen Register und Flags verändern. Damit verschafft er sich einen Überblick über die Wirkungsweise eines Prozessors; die Mikroschritte eines Maschinenbefehls werden einzeln dargestellt und durchlaufen.

Der auf der Diskettenrückseite mitgelieferte ASSYST-Assembler ist ein interaktiver Editor/Assembler für die Apple-II-Familie. Er ist lediglich als Lehrmittel zum Erlernen des Assemblierablaufs und für Hobby-Programmierer gedacht, die nur kurze Programme schreiben wollen. Die Assemblierzeiten sind entsprechend lang und schrecken professionelle Anwender ab: Für das 199 Zeilen lange Programm „Mini Assembler" benötigt der ASSYST sage und schreibe 9 Minuten und 53 Sekunden zum Laden und Assemblieren. (Zum Vergleich: der Assembler ASSESSOR, der auf der Begleitdiskette zum Assembler-Kurs „Apple-Assembler lernen, Bd. 1" von Dr. J. Kehrel mitgeliefert wird, erledigt die gleiche Aufgabe in 7 Sekunden!)
Für eigene Programme steht mit dem VISIBLE COMPUTER nur sehr beschränkt Platz zur Verfügung: Von Adresse $0800 bis $0D00 liegt 1K freier Benutzerspeicher.

Kurz und gut: Wer sich nach dem „Hineinschnuppern" eingehender mit Assemblerprogrammierung befassen will, wird VISIBLE COMPUTER bald beiseite legen. Unter diesen Umständen sollte man sich überlegen, ob man die DM 58,–, die VISIBLE COMPUTER kostet, nicht gleich besser investiert und sich einen professionellen Assembler zulegt.

Deutsch lernen mit dem Apple

Ein Computerprogramm zur Übung der Präpositionen

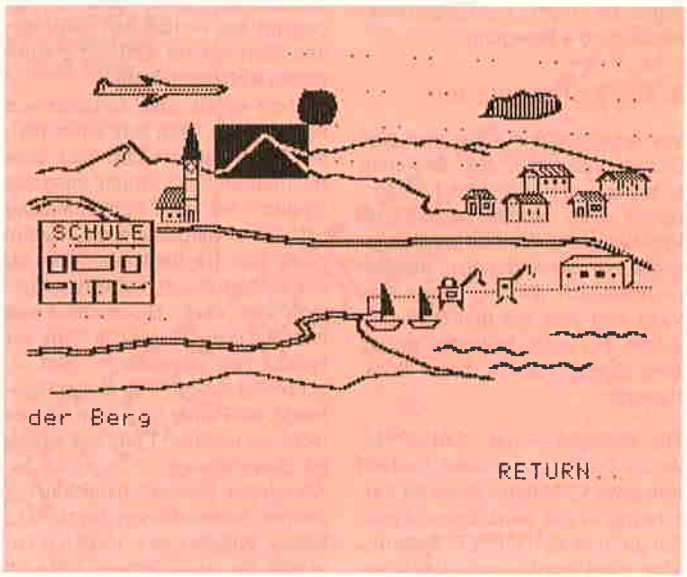
Der Verlag für Deutsch hat für Jugendliche und Erwachsene, die in der Grundstufe Deutsch als Fremdsprache lernen, das Computerprogramm „Morgens geht Fritz zur Schule" entwickelt. Es kann zusätzlich zu jedem Deutsch-Lehrbuch oder lehrbuchunabhängig verwendet werden. Das Programm dient zur Übung der wichtigsten deutschen Präpositionen mit Dativ und Akkusativ und ist lauffähig auf Apple II/II+/IIe mit 48K und Applesoft-ROM oder Language-Card. Für DM 160,– erhält man zwei einseitig bespielte, kopiergeschützte Disketten und eine 4seitige Anleitung. Das Programm kann ohne jede Programmierkenntnisse bearbeitet werden. Zur Lösung der gestellten Aufgaben muß der Schüler lediglich die Return-Taste und vorgegebene Zifferntasten drücken oder fehlende Präpositionen an gekennzeichneten Stellen einfügen. Nach jeder Antwort folgt sofort die Auswertung, fehlerhafte Eingaben werden korrigiert. Durch variierende akustische Untermalungen oder optische Kennzeichnungen ist die Korrektur besonders einprägsam.

Ablauf

Im ersten Abschnitt werden die vorkommenden Wörter anhand von Beispielgrafiken erklärt; danach werden die Pluralformen einiger Wörter gebildet. Der dritte Teil befaßt sich mit den wichtigsten Präpositionen. Funktion und Bedeutung der Präpositionen werden an bewegten Grafiken verdeutlicht. Im Übungsteil muß der Schüler aus einer Liste von Möglichkeiten selbständig die richtigen Präpositionen in vorgegebene Beispielsätze einfügen. Auch hier werden falsche Antworten sofort verbessert. Nach jeder Übung kann der Schüler wählen, ob er den Abschnitt nochmals wiederholen möchte oder zum nächsten Punkt übergeht.
Im Testteil, der sich auf Diskette 2 befindet, werden Beispielsätze gebildet, in die vorgegebene Präpositionen und Artikel eingesetzt werden müssen. Im Artikel-Quiz muß der Sprachschüler der vorgegebenen Präposition, dem Geschlecht und dem Numerus den korrekten Artikel zuordnen.

Fazit: Für deutschlernende Sprachschüler stellt das Computerprogramm sicher eine abwechslungsreiche, motivierende Möglichkeit dar, deutsche Präpositionen gründlich zu lernen. Für Interessenten stellt der Verlag eine kostenlose Demodiskette zur Verfügung, die die Möglichkeiten und Einsatzgebiete des Programms zeigt.

Bezugsquelle: Verlag für Deutsch, München



CADAPPLE

Ein CAD-System für den Apple II

getestet von Hans-Martin Eng

1. Überblick

CAD (Abkürzung für „computer aided design", d.i. computerunterstützte Konstruktion) ist in letzter Zeit in aller Munde, besser gesagt, in aller Rechner. CAD-Systeme laufen auf Großrechnern, Mini-Computern und Personal Computern. Alle, die im entferntesten mit Konstruktion zu tun haben, werden sich in Zukunft mit CAD befassen müssen.

Das CAD-System Entry-Level CADAPPLE wendet sich an Anfänger, die den Umgang mit dieser neuen Technik lernen möchten. Es läuft auf jedem Apple II, der über 64K RAM und *zwei* Diskettenlaufwerke verfügt. Außerdem werden ein Eingabegerät für x/y-Koordinaten (Koala Pad, Maus oder Joystick, notfalls auch die Tastatur) und ein Plotter benötigt. An Plottern werden (bisher) nur die HP-GL-Plotter, der DM-PL+-Plotter und der Apple-Plotter unterstützt.

Zum Lieferumfang des CADAPPLE gehören vier Disketten, eine Tonbandkassette und zwei Handbücher. Beigelegt ist noch ein Aufkleber, den man am oberen Rand der Tastatur anbringt. Über die obere Tastaturreihe werden verschiedene Menüfunktionen angewählt, der Aufkleber dient als Gedächtnisstütze für die Belegung.

2. Die Komponenten

Vor Arbeitsbeginn sollte man sich Sicherungskopien der Disketten anfertigen, z.B. mit dem Transferbefehl des Apple-Pascal-Filers. Von den vier Disketten werden jeweils nur zwei benutzt, nämlich grundsätzlich die sog. Work-Diskette und eine der drei Boot-Disketten. Für jeden Plottertyp gibt es eine eigene Version dieser Bootdisketten.

Die Kassette – die „CADAPPLE Audio-Tour" – mit einer Laufzeit von etwa 45 Minuten dient als Einführung in die prinzipiellen Funktionen des CADAPPLE-Systems. Man kann jedoch auch gleich anfangen und Band 1 des Handbuchs, das Tutorial, durcharbeiten. Dort wird all das vertieft, was in der Audio-Tour nur oberflächlich berührt wurde. Als besonders wertvoll erweisen sich die Aufgaben, die nach jedem Sinnabschnitt gestellt werden und der Überprüfung des gerade erworbenen Wissens dienen. Der Benutzer kann anhand eines Lösungsschlüssels schnell die Richtigkeit seiner Antworten überprüfen (besser der Kursleiter, sonst wird der Lernende zum „Spicken" verführt).
Im 45seitigen Reference Manual sind alle Befehle aufgeführt. Außerdem sind Anweisungen zum Kopieren der Systemdisketten (dies ist uneingeschränkt möglich) und zur Einstellung der Super Serial Card für den Anschluß des Apple-Plotters enthalten. Die Erläuterungen sind knapp, aber ausreichend, alle Menüoptionen werden genannt und kurz in ihrer Funktion erläutert.

3. Bedienung

3.1. Erste Schritte

Mit der ersten Anweisung der Kompaktkassette wird man (auf englisch) aufgefordert, das System zu starten. Schritt für Schritt wird der Benutzer dann durch die verschiedenen Menüs geführt. Das beginnt bei einfachsten Schritten und führt hin zur Erstellung einer ersten einfachen Grafik.
Bei mir ergab sich da gleich ein Problem: Ich hatte kurz zuvor meine Super Serial Card aus dem Rechner entfernt. Prompt stieg das System mit der Fehlermeldung „No Serial Card found, you cannot plot" aus. Da fragt man sich, ob das System derart restriktiv ausgelegt sein muß. Glücklicherweise hatte ich ja meine serielle Karte zur Hand. Reingesteckt in Slot 2, schon lief CADAPPLE an (wohlgemerkt, ein Plotter hing nach wie vor nicht am anderen Ende der seriellen Datenleitung).
Als „Input Device" hatte ich nur meinen mittelmäßigen Joystick zu bieten. Wegen des schlechten Zustands der Potentiometer hatte ich erhebliche Schwierigkeiten, einen Punkt exakt zu setzen, obwohl CADAPPLE Punkte nur in einem vorgewählten Raster setzt. Es empfiehlt sich also, einen guten Joystick oder besser eine Maus oder ein Koala Pad als Eingabeelement zu verwenden.

3.2. Das Tutorial

Nun vertraute ich mich den ersten Kapiteln des englischsprachigen Tutorials an. Das beginnt wieder beim Punkte Null, was nichts schadet; man fragt sich dann jedoch nach dem Sinn der Kompaktkassette. Mir ist die Führung durch das Tutorial irgendwie sympathischer. Dort sieht man anhand von kleinen Abbildungen, wie das Bild auszusehen hat, das man auf den Monitor zaubern soll. Mich persönlich stört es nicht allzusehr, daß das Tutorial in Englisch geschrieben ist. Die deutschen Vertreiber sollten jedoch zumindest das Tutorial ins Deutsche übersetzen lassen, denn man kann nicht von jedem „CAD-Azubi" verlangen, daß er perfekt Englisch versteht.

3.3. Die Menüführung

Die Menüführung ist sehr übersichtlich gehalten. Aus jedem Menü führen die gleichen Befehle ins übergeordnete Menü zurück, die einzelnen Optionen sind immer auf einer Textseite aufgeführt. So kann man einfach und effektiv die gewünschten Programmteile aufrufen und an der Grafik entsprechende Modifikationen vornehmen. Die zu einem CAD-System gehörenden Grundfunktionen bietet CADAPPLE in jedem Fall (Zusammenfassung bestimmter Grafikelemente zu Makros, Ellipsen, Rechtecke, Linien zeichnen, füllen und löschen etc.)

4. Ergebnisse

4.1. Ausgabe

Da ich leider keinen Plotter besitze, kann ich nicht mit Beispielausdrucken dienen. Die Ausgabe von Grafiken auf einem Matrixdrucker ist nicht vorgesehen. Folgende Plotter werden laut Handbuch unterstützt: Der Apple-Plotter, die HP-GL-Plotter (das sind Plotter der Firma Hewlett Packard, die über standardisierte Grafikbefehle angesteuert werden können) und der DM/PL+-Plotter.

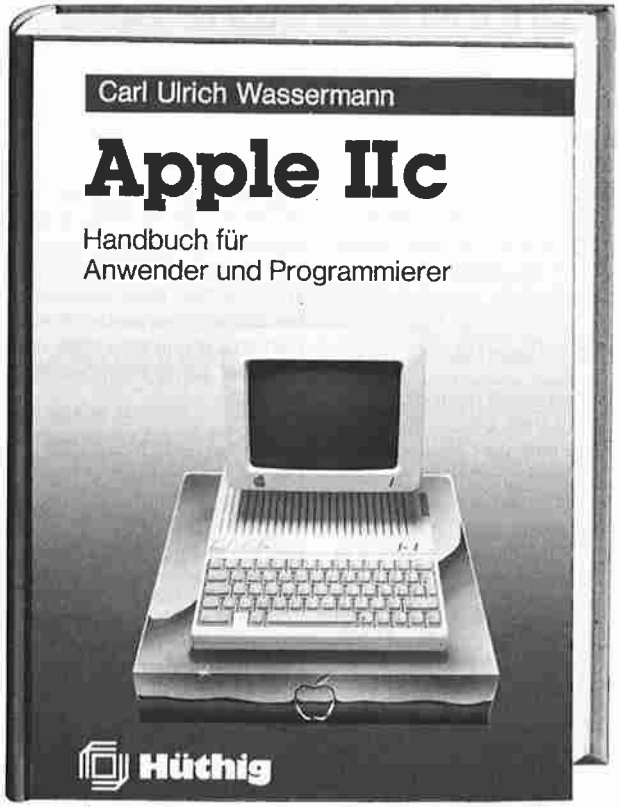
4.2. Bearbeitungsgeschwindigkeit

Man sollte von CADAPPLE keine Geschwindigkeitsrekorde erwarten. Dem steht schon das Betriebssystem entgegen – CADAPPLE ist in Apple-Pascal geschrieben. Aber wieso auch nicht, man soll schließlich nur an die Grundbegriffe des CAD herangeführt werden und nicht einen neuen Mittelklassewagen konstruieren. Im übrigen bin ich davon überzeugt, daß professionelle CAD-Systeme in Hochsprachen geschrieben sind. Bei den geringen Stückzahlen und den hohen Qualitätsanforderungen wäre es kaum wirtschaftlich, CAD-Software in Assembler zu entwickeln. Die Verwendung von Apple-Pascal für CADAPPLE ist also durchaus legitim.

5. Bewertung

Betrachtet man die Zielgruppe von CADAPPLE (Schulen, Hochschulen und ausbildende Betriebe), so muß man uneingeschränkt anerkennen, daß das gesteckte Ziel erreicht wurde: Mit geringsten Mitteln erhält der Lernende eine umfassende Einführung in den Umgang mit CAD-Systemen. Ob CADAPPLE sich allerdings in der harten Realität eines normalen Konstruktionsbetriebs behaupten kann, wage ich zu bezweifeln.
Positiv zu bewerten ist die Tatsache, daß mit geringen Mitteln schon gute Ergebnisse zu erzielen sind. Am didaktischen Aufbau des Tutorials gibt es nichts auszusetzen, nur sollte es auch auf Deutsch erhältlich sein.
So bleibt als Testergebnis festzuhalten: Für den Ausbildungsbereich aufgrund relativ geringer Kosten (CADAPPLE kostet DM 298,–) und hoher Benutzerfreundlichkeit gut geeignet. Voraussetzung ist jedoch das Vorhandensein von genügend Apple-Arbeitsplätzen, die mit Plottern, zwei Laufwerken (oder einem Laufwerk und einer Harddisk) und Joysticks (bzw. Maus oder Koala Pad) ausgerüstet sind.
Der einzige Wermutstropfen ist, daß die Zeichenergebnisse nicht auf den gängigen Matrixdruckern ausgegeben werden können. Somit dürfte CADAPPLE für den Privatmann ungeeignet sein, denn ein Plotter ist und bleibt eine ziemliche Geldausgabe.

FACHBUCHTIP Hüthig



Apple IIc

Handbuch für Anwender und Programmierer

von Carl-Ulrich Wassermann

1985, 324 S., zahlr. Abb., kart., DM 35,—
ISBN 3-7785-1157-2

Wenn Sie die Leistungsfähigkeit ihres Apple IIc bisher noch nicht ausschöpfen konnten, brauchen Sie dieses Buch.

Leicht verständlich, trotzdem ausführlich wird die Sprache Applesoft BASIC dargestellt. Eine Vielzahl von Programmen, die speziell auf den Apple IIc zugeschnitten wurden, zeigen die Wirkung der einzelnen Befehle bis zu Programmiertechniken für Diskettenzugriff, Maus und andere Anwendungen. Apple IIc Maschinensprache, Programmeingabe und -kontrolle mit Monitorbefehlen in PASCAL und FORTH geben dem Leser eine Basis für die Programmierung des Apple IIc in anderen Sprachen.

Die Konfigurierung der seriellen Ports und weitere spezifische Beschreibungen des Computers ermöglichen die gezielte Kontrolle über die Maschine.

Mehr als 90 PEEK-, POKE-und Maschinenprogrammadressen, Speicheraufteilung und weitere Tabellen sind für den Programmierer eine konzentrierte, wichtige Grundlage.

BESTELLCOUPON

Bitte ausfüllen und an Hüthig Vertriebsservice, Postfach 102869 · 6900 Heidelberg schicken.

Buchtitel _____

Name _____ Ort _____

Straße _____ Unterschrift _____

Neue Firmware für Ehring-V80-Karte

von Dagmar Berberich

Die V80-Zeichenkarte der Fa. Ehring Elektronik ist eine weitverbreitete Karte, die ein sehr ruhiges und scharfes Bild erzeugt. Leider weist die Karte in der Firmware viele Schwächen auf. Aus diesem Grund hat Michael Bos eine neue Firmware für die V80-Karte entwickelt, die über die Kölner Firma Weber Computertechnik zu beziehen ist. Die neue Firmware-Version 3.04 ist etwa viermal so lang wie die Altversion 2.2. Vor dem Einsatz der neuen Version muß die V80-Karte hardwaremäßig modifiziert werden, da die Speicherkapazität des Firmware-EPROMs von 2K auf 4K verdoppelt werden muß. Dieser Umbau wird beim Kauf der neuen Firmware kostenlos durchgeführt.

Achtung:
Die Firma Ehring Electronic in Duisburg ist nach unseren Recherchen seit dem 14.11.1986 nicht mehr aktiv. Ob und wer die V80-Karte jetzt noch herstellt und/oder ausliefert, konnte nicht ermittelt werden. Die nachfolgende Beschreibung basiert auf der Bedienungsanleitung und ist damit kein Testbericht!

1. Lieferumfang

Zum Preis von DM 125,– erhält der V80-Besitzer die neue Firmware, Version 3.04, ein 53seitiges, deutschsprachiges Handbuch, eine Demodiskette und eine umschaltbare Hilight-Lolight-Modifikation. Der notwendige Umbau an der V80-Zeichenkarte wird kostenlos durchgeführt.

2. Manual

Das mitgelieferte, deutschsprachige Handbuch behandelt auf 53 Seiten die Neuheiten und Veränderungen der Firmware 3.04 gegenüber der Altversion 2.2 und erläutert die Handhabung der neuen Befehle und Eigenschaften. Im umfangreichen Anhang werden die Funktionstasten für den Apple II+ und IIe, die Befehle für neue Tastencodes, die Direktkommandos, alle Control-Funktionen und die Escape-Funktionen der Eingabe (über Tastatur) und Ausgabe (vom Programm aus) tabellarisch aufgelistet. Dabei werden alle Tabellen zur leichteren Benutzung zweimal ausgegeben: einmal nach Funktionstasten, dann nach Funktionsgruppen sortiert.
Ein Stichwortverzeichnis am Ende des Handbuchs hilft dem Benutzer, schell die gewünschte Information zu finden.

3. Demodiskette

Die Demodiskette dient dazu, sich bereits vor dem Umbau der eigenen Karte über die Leistungsfähigkeit der neuen Firmware zu informieren. Deshalb benötigen die Demo-Programme nur die alte V80-Karte in Slot 3 des Computers. Die neue Firmware wird als RAM-Version mitgeliefert. Sie ist aus diesem Grund allerdings nur eingeschränkt lauffähig, unter UCSD-Pascal oder CP/M z.B. läuft die RAM-Version nicht.
Die Demodiskette wird einfach gebootet, dann erscheint ein Programmenü, aus dem die Beispielprogramme ausgewählt und gestartet werden können. Zur Auswahl stehen Demos zu Applesoft-, Control-, Escape-, Buffer- und Tastatur-Befehlen, ein Demo zur Insert-Line-/Delete-Line-Funktion und zur Geschwindigkeit sowie ein „Mammutdemo", das einen automatischen Durchlauf durch alle Demoprogramme erlaubt (der immerhin über eine halbe Stunde dauert!).
Mit dem Merlin-Assembler geschrieben wurden das Source- und Objektcode-File des Assemblerprogramms, das im Demo „Geschwindigkeit" benutzt wird.
Unter dem Stichwort EPROMINHALTE befinden sich außerdem Firmware-Variationen für 3 verschiedene Rechnersysteme auf der Demodiskette.

4. Unterschiede zur Version 2.2

Bei der Firmware-Version 3.04 wurden nicht nur Eigenschaften neu implementiert, sondern auch Eigenschaften des Vorgängermodells verändert.

Geschwindigkeit
Alle Routinen der neuen Firmware laufen mit maximaler Geschwindigkeit ab. Die Zeichenausgabegeschwindigkeit lag bei der Version 2.2 noch bei 2700 Zeichen/s, die Version 3.04 erreicht nahezu 6000 Zeichen/s. Diese Zeiten geben nur die reine Ausgabegeschwindigkeit der V80-Karte und nicht die des Programms an; in Applesoft-BASIC kann man mit einer Geschwindigkeitserhöhung um den Faktor 1,5 bis 2 rechnen. Die Routinen zum Löschen einer Zeile oder eines Bildschirms wurden direkt implementiert und sind etwa dreimal schneller als die Routinen der Version 2.2.

Softswitch und Zeichensatz
Der einmal angewählte Status des Softswitch bleibt auch nach der Eingabe eines Zeichens bestehen; dies gilt auch bei einem Warmstart der 80-Zeichenkarte nach einem Reset.

Normal-Invers-Funktion
Zusätzlich zu den bisherigen Funktionen Control-Q und Control-R wurden Control-N und Control-O zum Umschalten zwischen Normal- und Invers-Modus eingeführt. Damit sind diese Funktionen unter CP/M ohne Änderungen verfügbar.

Control-S, Control-Z
Um Schwierigkeiten unter Wordstar zu vermeiden (dort setzt Ctrl-S den Cursor um ein Zeichen nach links), funktioniert die Stop-List-Funktion Control-S nur noch unter Applesoft-BASIC bzw. unter CP/M und UCSD-Pascal, von wo sie direkt unterstützt wird.
Control-Z wurde durch einen leistungsfähigeren Befehl ersetzt.

Kleinschrift
Mit der neuen Firmware ist die Eingabe von Kleinschrift auch bei einem Apple mit den Original-Monitor-ROMs möglich, bei dem bisher alle Kleinbuchstaben vom Monitor-Programm automatisch in Großbuchstaben verwandelt wurden.

Initialisierung
Die Version 3.04 sollte nur mit dem „PR#"-Kommando und nicht über das „IN#"-Kommando initialisiert werden, sonst werden 2 zufällige Zeicheneingaben erzeugt.

5. Neuheiten der Version 3.04

In der neuen Firmware wurde eine ganze Reihe von Befehlen und Eigenschaften neu implementiert.

Erzeugung von nicht vorhandenen Tastencodes
Da nicht alle Tastaturen alle Zeichen des ASCII-Zeichensatzes erzeugen können, wurde in die Firmware die spezielle Control-Funktion <Ctrl-N> eingebaut, mit der man seltene Tastencodes erzeugen und der V80-Karte Befehle geben kann.

Groß- und Kleinschrift
Über <Esc> wurde eine Möglichkeit geschaffen, Groß- und Kleinschrift für ein oder alle folgenden Zeichen (auch für über Ctrl-N erzeugte Sonderzeichen, z.B. „Ä") anzuwählen.

Funktionstasten
Die Version 3.04 bietet über 30 verschiedene Funktionstasten, mit denen man häufig benutzte Kommandos schneller bzw. komfortabler eingeben kann.

Statuszeile
In Zeile 25 kann eine Statuszeile eingeblendet werden, die anzeigt, welcher Zeichensatz aktiv ist, ob bestimmte Eigenschaften der V80 ein- oder ausgeschaltet sind, ob Groß- oder Kleinschreibung angewählt wurde usw.

Insert Line/Delete Line
Diese zusätzlich eingeführte Funktion erlaubt höhere Verarbeitungsgeschwindigkeiten bei der Texteingabe von Programmen wie Wordstar oder Turbo-Pascal.

Bildschirminhalt laden/speichern
Der gesamte Inhalt des Bildschirmspeichers wird in einem Puffer abgespeichert und kann von dort aus wieder geladen werden; Abspeichern und Laden des Bildschirminhaltes auf bzw. von Diskette ist dadurch möglich.

Tabulator
<Ctrl-I> setzt den Cursor auf die nächste, durch 8 teilbare Position (horizontales Tabulieren).

Apple-IIe-Kompatibilität
Die V80 ist durch eine Anzahl neuer Befehle in fast allen Funktionen softwarekompatibel zur 80-Zeichenkarte des Apple IIe.

Restricted Case Modus
Da Applesoft-BASIC nur Befehle in Großschrift akzeptiert, werden alle Zeichen in Großbuchstaben umgewandelt. Nicht betroffen sind lediglich Buchstaben innerhalb von Anführungszeichen.

Abschalten der V80
Die Karte kann von der Tastatur oder über das Programm (jedoch nur unter Applesoft) durch den Befehl <Ctrl-U> ausgeschaltet werden.

BASIC-Befehle
Auf der V80 funktionieren die BASIC-Befehle HOME, TAB, HTAB, LIST und PRINT einwandfrei.

Softswitch und Zeichensatz
Softswitch und Zeichensatz sind

über die Tastatur oder über eine Zeichenausgabe vom Programm aus steuerbar; die Control-Funktionen bleiben über die Tastatur voll erhalten.
Cursor-Modus
Der Cursor-Modus (Flash, Normal, Block- oder Zeilencursor) kann über die Tastatur oder vom Programm aus geändert werden.
Key-Click
Optional kann bei jedem Tastendruck ein „Klick" als akustisches Kontrollsignal ausgegeben werden.
Bell on error
Anstelle des normalen „Beep" kann ein anders klingendes Alarmsignal ausgegeben werden, wenn die V80 aus einem Programm ein illegales Kommando empfängt.
40-Zeichen-Emulation
Nach dem Umschalten in den 40-Zeichen-Emulationsmodus reagiert die Karte, als hätte sie horizontal nur 40 Zeichen. Dadurch sind Programme wie FID ohne Modifikationen lauffähig.
Esc-Modus
Ein Karo signalisiert, daß sich die V80 im Esc-Modus befindet.
Kartenerkennung
Die V80 mit der Firmware 3.04 ist als 80-Zeichen-Firmware-Card mit der Device-Signatur $81 ausgewiesen. Dadurch wird sie von allen Betriebssystemen erkannt und richtig benutzt. Durch die Kennzeichnung als Firmware-Card wird auch der Input-Status unterstützt; damit wird per Programm festgestellt, ob eine Taste gedrückt wurde oder nicht.

6. Kompatibilität

Die neue Firmware läuft trotz der umfangreichen Änderungen mit fast allen Programmen, die auch mit der alten Firmware-Version 2.2 liefen, und wird von allen Betriebssystemen erkannt. AppleWriter, Multiplan, Magic Window und andere Programme, die die V80 benutzen, laufen fehlerfrei. Wordstar und Turbo-Pascal müssen auf die zusätzlichen Befehle der neuen Firmware angepaßt werden. Programme, die für die 80-Zeichenkarte des Apple IIe geschrieben wurden, aber die optional eingebauten 64K RAM dieser Karte nicht benutzen und die IIe-Karte nicht hardwaremäßig ansprechen, laufen auch auf der V80-Karte mit neuer Firmware.
Anpassungen sind bei allen Programmen erforderlich, die Informationen aus dem Bildschirmspeicher zurücklesen, weil der Zugriff auf den Video-Schirm bei der neuen Firmware grundlegend verändert wurde.

MUSIX32

Ein Musik-Editor für den Atari ST

getestet von Dieter Geiß

Bei MUSIX32 handelt es sich um einen speziell für Atari-ST-Computer entwickelten Musik-Editor. Mit ihm können einfache, bis zu dreistimmige Melodien komponiert und in einer solchen Form abgespeichert werden, daß sie in eigene Programme eingebunden werden können.

1. Lieferumfang

Für DM 89,– erhält man eine stabile Box, die eine einseitig formatierte Diskette und eine 40seitige Anleitung enthält.
Auf der Diskette findet man neben dem eigentlichen Programm eine große Anzahl von fertigen Musikstücken, die man sofort abspielen kann, sowie einige Demoprogramme. Ebenfalls zum Lieferumfang gehört eine eigens entwickelte RAM-Disk. Sie ist recht einfach gehalten und testet z.B. nicht ab, ob man beim Installieren zuviel RAM angibt. Dies ist aber im Handbuch vermerkt. Besitzt man noch keine RAM-Disk, so kann man die mitgelieferte benutzen, um in den Genuß des Geschwindigkeitsvorteils zu kommen.

Die Originaldiskette ist kopiergeschützt und muß sich bei jedem Programmstart in Laufwerk A befinden. Dieser Umstand ist vor allem beim Installieren auf einer Festplatte ein enormes Hindernis. Im Handbuch wird zwar auf das leidige Thema Software-Piraterie hingewiesen, aber daß es auch anders geht, zeigt beispielsweise das Software-Haus ADI aus Karlsruhe: Für das relationale Datenbanksystem ADIMENS ST, das um einiges teurer ist als MUSIX32, hat es keinen Kopierschutz installiert. Ohne Kopierschutz ist ein Arbeiten mit der Festplatte dann problemlos möglich. Zu Testzwecken wurde eine Kopie erstellt. Diese lief einwandfrei, so daß ich mich natürlich frage, was der Kopierschutz dann überhaupt bringen soll.

Startet man das Programm, so findet man bald heraus, daß es nicht mit jeder Konfiguration läuft. So ist beispielsweise nicht möglich, ein Bild mit GEM Draw zu entwerfen und danach ein Musikstück mit MUSIX32 zu komponieren. MUSIX32 weigert sich, die Biene in einen Pfeil umzuwandeln, um die Arbeit aufzunehmen. Es ist das erste mir bekannte Programm, das nicht läuft, wenn die Gerätetreiber für Metafiles etc. über das bekannte GDOS.PRG im Auto-Ordner installiert sind. Bevor ich MUSIX32 anwenden kann, muß ich also neu booten, wobei sich im Auto-Ordner nicht das GDOS.PRG befinden darf. Dann kann ich allerdings nicht mehr GEM Draw, GEM Paint oder GEM Write benutzen – eine unverständliche Einschränkung, da von MUSIX32 keine Metafiles erzeugt werden. Es gibt daher eigentlich keinen Grund, bei angeschlossenen Treibern den Dienst zu versagen. Leider kann man auch nicht auf einem Farbmonitor arbeiten, da das gesamte Kontrollfeld sonst nicht auf eine ganze Bildschirmseite passen würde.

2. Die Arbeitsfläche

Noch bevor man zur Arbeitsfläche gelangt, hat man die Möglichkeit, eine Sprache auszuwählen: Deutsch oder Englisch. Danach erscheint ein grafisch recht ansprechend gemachtes Arbeitsblatt, das im wesentlichen aus drei Teilen besteht. In der obersten Zeile kann man unter acht verschiedenen Notenzeichen, vier Vorzeichen, fünf Pausenzeichen und vier weiteren Sondersymbolen wählen. Der Mauszeiger kann diese 21 Formen annehmen. Die Auswahl erfolgt durch Zeigen auf ein Symbol, ohne daß man es anklicken muß.
Im mittleren Drittel befinden sich zwei Notenzeilen. Die obere Zeile zeigt den Violin- und den Baßschlüssel sowie die ausgewählte Tonart und den Takt an. In der darunterliegenden Zeile können eigene Noten eingegeben werden, und zwar getrennt für jeden der drei ansprechbaren Tonkanäle. Beim Eingeben der Noten wird automatisch gescrollt, wenn ein Überlauf stattfindet. Die Entfernung zweier Taktstriche voneinander richtet sich nach dem Taktmaß, das zwischen Ein-Achtel- und Acht-Viertel-Takt beliebig gewählt werden kann.
Das untere Drittel des Arbeitsblattes zeigt ein ausführliches Funktionsmenü, das keinerlei sprachspezifische Zeichen, sondern Symbole enthält, die alle sehr hübsch dargestellt werden. Insgesamt stehen 36 Funktionen zur Verfügung. Zu Ihnen gehören:

– Blättern und Scrollen im Musikstück (vorwärts und rückwärts)
– Einstellen der Ablaufgeschwindigkeit (Symbol Hase und Igel)
– Einstellen des Taktmaßes in 1/16-Schritten
– Anwählen der drei Tonkanäle (exklusiver Ausschluß)
– An-/Ausschalten des Eingabe-Echos
– Starten und Beenden des Abspielvorgangs
– Einstellen des Kammertons A
– Ausdrucken von Musikstücken
– Ausschneiden, Kopieren und Einfügen
– Laden und Speichern von Musikstücken
a) in MUSIX32-Form
b) in XBIOS-Form
– Ein-/Ausschalten einer Hilfsfunktion
– Löschen des gesamten Stückes
– Verlassen von MUSIX32

3. Editieren von Musikstücken

Das Arbeiten geht recht flott von der Hand. Das liegt vor allem an den schnellen Scroll-Routinen, die beim Blättern und Markieren zum Tragen kommen. Es gibt die Möglichkeit, während des Editierens alle oder nur einen Tonkanal anzeigen zu lassen. Die erste Art zeigt den Zusammenhang aller drei Ton-

kanäle, ist aber manchmal etwas unübersichtlich. Die zweite Art ist zwar übersichtlich, aber der Zusammenhang fehlt. Der Benutzer kann wählen, welche Möglichkeit ihm angenehmer ist.
Das Löschen kann notenweise über den DEL-Pfeil erfolgen oder über die Blockoperationen. Mit diesen kann man auch in einen Puffer kopieren, so daß damit Wiederholungen möglich werden.

4. Abspielen von Musikstücken

Beim Abspielen hat man drei verschiedene Möglichkeiten:
– Abspielen des gesamten Stücks
– Abspielen von der aktuellen Position bis zum Ende
– Abspielen des momentanen Bildschirminhalts

Für Tonkanal A kann dabei ein Schalter gesetzt werden, worauf sich diese Stimme anhört wie ein scharf angeschlagenes Klavier. Das ist die einzige Möglichkeit, den Sound zu verändern. Sonst hören sich alle Stimmen immer gleich an. Ein Hüllkurvenformer wird voraussichtlich Anfang 1987 als Zusatzmodul angeboten.
Die Abspielgeschwindigkeit kann in neun Stufen eingestellt werden (über Hase und Igel). Allerdings muß man das Stück dann jedesmal neu starten, bevor man eine Änderung hört.

5. Einbindung in eigene Programme

Das hervorstechendste Merkmal von MUSIX32 ist wohl die Fähigkeit, die Musikstücke derart abzuspeichern, daß sie über den XBIOS-Aufruf „Dosound" in eigene Programme übernommen werden können. Die Programmierung selbst ist sehr einfach und wird an drei Beispielen sauber dargelegt. Die Beispiele sind in verschiedenen Programmiersprachen geschrieben, namentlich Pascal (ST Pascal plus), C und BASIC (GFA-BASIC). Bis jetzt kenne ich kein anderes Produkt, das diese Fähigkeit anbietet.

6. Nachteile von MUSIX32

Hier sei noch einmal der Kopierschutz erwähnt, der das Arbeiten auf einer Festplatte erschwert. Auf die Inkompatibilität im Zusammenhang mit GDOS wurde ebenfalls bereits hingewiesen.
Auf der Rückseite des Handbuchs wird von einem professionellen Komponierwerkzeug gesprochen. Darunter stelle ich mir allerdings etwas anderes vor. Das wichtigste ist wohl der Klang, der sich – zumindest im Augenblick – fast gar nicht verändern läßt. Dadurch klingen alle Musikstücke irgendwie gleich. Stücke von „The Music Studio", das nur wenig teurer ist, sind da vielseitiger. Dort kann man zusätzlich den Rauschgenerator benutzen und die Wellenformen für verschiedene Instrumente selbst einstellen.(Sogar Hihat und Snare Drum werden simuliert, so daß die Stücke wesentlich besser und variationsreicher klingen.)

Ein weiterer Nachteil besteht darin, daß in Tonkanal A und C zwei Noten mit der gleichen Tonhöhe nicht unterschieden werden. Sie verschmelzen zu einer einzigen Note, die entsprechend länger ertönt. Bei realen Instrumenten dagegen sind die einzelnen Anschläge immer zu hören. Man könnte diesen Umstand umgehen, indem man Staccato-Noten eingibt, aber die existieren auf MUSIX32 leider nicht. Abhilfe würde das zusätzliche Einfügen von kurzen Pausen nach jeder Note schaffen, aber das artet in viel Arbeit aus. Die Möglichkeit, Triolen einzugeben, sollte in einem Programm, das sich professionell nennt, nicht fehlen – auch wenn sie in Musikstücken nicht so häufig vorkommen. Das gilt auch für Mehrfachpunktierungen (viertel Note + achtel Note + sechzehntel Note).
Das Ausdrucken der Musikstücke ist zwar eine hübsche Option, jedoch werden die Noten so ausgegeben, wie sie auf dem Bildschirm erscheinen, nämlich einzeln und ohne jegliche Verbindungen oder Balken. Die Hilfspünktchen, die dem besseren Plazieren der hohen oder tiefen Noten auf dem Bildschirm dienen, werden auf dem Papier mit abgedruckt; dies sollte man unterdrücken können. Eine Verbindung zur MIDI-Schnittstelle, wie z.B. bei „The Music Studio" fehlt ebenfalls.

Fazit

MUSIX32 eignet sich vor allem für diejenigen, die ihre eigenen Programme auf einfache Art mit Musik unterlegen wollen. Diese Musik genügt jedoch keinen großen Ansprüchen, da es praktisch keine Klangvariationen gibt. Die nächste Version könnte diese Mängel beheben. Die Bezeichnung „professionell" scheint mir allerdings nicht ganz gerechtfertigt, vergleicht man MUSIX32 beispielsweise mit „The Music Studio". Die Grafik ist zwar recht ansprechend gemacht, aber bei einem Musikprogramm sollte die Musik im Vordergrund stehen. Der Atari Soundchip wurde hier bei weitem nicht voll ausgenutzt.

DISK40

Disketten-Organisationsprogramm für Apple II+, IIe oder IIc

von Hermann Seibold und Dipl.-Ing. Udo Marin, 1986, Programmdiskette mit Anleitung, DM 38,–

DISK40 entstand aus der Analyse bestehender Kopierprogramme und vereint in sich eine Vielzahl von Möglichkeiten, die sich als nützlich erwiesen haben. Durch eine einfach zu bedienende Menüführung können DOS-3.3-Disketten umfangreich bearbeitet oder kopiert werden.
Zu den vielfältigen Möglichkeiten des Programms zählen u. a.:

– Tabellarische Ausgabe der Diskettenbelegung
– Ordnen des Catalogs
– „Undelete"n von versehentlich gelöschten Dateien
– Vergleichen von Disketten, Dateien oder der DOS-Spuren
– Kopieren von Disketten, Dateien oder DOS-Spuren
– Formatieren von Daten-Disketten
– Erweitern auf 40 Spuren bei bestehenden 35-Spur-Disketten
– Ändern des Boot-Programms
– File-Editor zum Editieren von Disketten-Dateien
– Komfortabler Sektor-Editor für Hex- und ASCII-Darstellung
– VTOC-Editor, z. B. zur Freigabe der DOS-Spuren

Schon nach wenigen Minuten können, dank der ausführlichen Beschreibung, Disketten nach eigenen Wünschen modifiziert oder Daten nach einem Disk-Crash wieder gerettet werden.

Hüthig Software Service · Postfach 102869 · Heidelberg 1

Arne Schäpers

Bewegte Apple-Graphik

DOS Toolkit-Erweiterungen

1985, 305 S., zahlr. Abb.,
kart., DM 58,–
ISBN 3-7785-1150-5

„Bewegte Grafik, Apple DOS Toolkit Erweiterungen" wendet sich als lehrbuchhafter Kurs an alle, die professionelle hochaufgelöste Grafiken auf dem Apple erzeugen wollen. Der erste Teil beginnt mit einem Abriß des Aufbaus der HGR-Seiten aus der Sicht des Programmierers. Danach wird das Programm HRCG (Hires Character Generator, Apple, Inc.) eingehend analysiert, und sinnvolle Ergänzungen werden vorgestellt. Schrittweise wird die Nutzung des HRCG erarbeitet bis hin zur beliebigen Bewegung eines statischen Objekts auf einer der HGR-Seiten.

Der zweite Teil baut auf dem ersten auf und führt über die Definition mehrerer Objekte und simultaner Bewegung hin zu einem Arcade-Spiel, das für die meisten käuflichen Action-Spiele in der meisterhaften Grafik als Vorbild dienen kann. Zielgruppe sind Programmierer, die mit Basic keine großen Schwierigkeiten mehr haben und zumindest über praktische Grundkenntnisse in 6502-Assembler verfügen sollten.

Dr. Alfred Hüthig Verlag
Im Weiher 10
6900 Heidelberg 1

Atari-Produkte

ST Pascal plus

2. Auflage 1986, 403 S., kart., mit Programmdiskette; CCD, Eltville (Vertrieb über Atari, Raunheim)

Diese auf den Atari ST zugeschnittene Pascal-Version umfaßt Editor, Compiler und Linker und ist ähnlich preiswert wie Turbo-Pascal. Was die normalen Pascal-Befehle anbelangt, so handelt es sich um ein erweitertes Standard-Pascal mit zusätzlichen Datentypen (z.B. Byte; funktioniert jedoch nicht beim Diskettenzugriff!), Befehlen (z.B. Links- und Rechtsverschiebung ShL und ShR) und Konstrukten (z.B. Loop – Exit). Die Ausführungsgeschwindigkeit des compilierten Programms ist teils höher (z.B. bei Integer-Zahlen) und teils niedriger (z.B. bei Real-Zahlen) als bei GFA-BASIC.

Es wurden alle GEMDOS-, BIOS- und XBIOS- sowie die meisten GEM-, VDI- und AES-Befehle implementiert, so daß ST Pascal plus eine echte Alternative zur C-Programmierung darstellt. Um sinnvoll arbeiten zu können, muß man alle Module auf eine RAM-Disk (ca. 500K) kopieren (Driver wird mitgeliefert). Dann reduziert sich das Compilieren und Linken kleinerer Programme auf wenige Sekunden. Insgesamt hinterließ dieses 68000-Pascal einen sehr positiven Eindruck (komfortabler Editor, unkompliziertes Compilieren und Linken per Knopfdruck usw.). Ein Mangel ist jedoch die Tatsache, daß ein Array nur insgesamt 32K einnehmen kann (bei Long Integer mit 4 Bytes je Zahl z.B. Fehlermeldung bei Long Array[1..8192]).

GST Macro-Assembler

1985, ca. 200 S. (kapitelweise paginiert), Loseblattwerk, mit Programmdiskette, GST, Cambridge (Vertrieb über Atari, Raunheim)

Dieses Programmpaket besteht aus Editor, Assembler und Linker. Der erzeugte Objektcode enthält normalerweise eine Relokationstabelle, doch kann man auch reine Maschinenprogramme erzeugen, die dann allerdings von vornherein relokativ sein müssen (vgl. GST-Assembler-Programmbeispiel im Peeker 2/86 zum Einbinden in GFABASIC). Das Manual ist sehr ausführlich und erstaunlich präzise bei Begriffsdefinitionen (z.B. bei dem mehrdeutigen „#", s.S. 34 im Assembler-Teil).

Die einzelnen Module lassen als solche keine Wünsche übrig, wenn Sie nur nicht so schrecklich langsam wären. Offenbar wurde das gesamte Paket in C geschrieben. Selbst wenn man eine RAM-Disk einsetzt, ist der Wechsel zwischen Editor, Assembler und Linker sowie das Assemblieren selbst extrem behäbig, insbesondere wenn man zur Kontrolle auf den Bildschirm listet (mit „lst: con:").

Assembler-Praxis auf Atari ST

von Roland Löhr, 1986, 258 S., kart., DM 59,-, Tewi-Verlag, München

Eine nützliche Einführung in die 68000-Programmierung. Kenntnisse eines anderen Prozessors werden allerdings vorausgesetzt, da vieles nur angedeutet wird. Zudem sind die Angaben nicht immer so präzise, wie man es sich wünschen würde (Wann Word? Wann Long? Welche Register bleiben bei welchen Systemaufrufen unverändert? usw.)

Das C-Buch

von H. Herold und W. Unger, 1986, ca. 550 S. (kapitelweise paginiert), kart., DM 79,-, Tewi-Verlag, München

Dieses Buch dürfte eine der besten Einführungen in C sein, die sich auch für Anfänger eignet, für die C die erste Programmiersprache darstellt. Dank der hervorragenden didaktischen und satztechnischen Aufbereitung mit vielen eingestreuten Flußdiagrammen und sonstigen Schaubildern macht es Freude, mit diesem Werk zu arbeiten.

M68000-Familie

Teil 1: Grundlagen und Architektur, von Werner Hilf und Anton Nausch, 1984, 568 S., kart., DM 79,-; Teil 2: Anwendung und 68000-Bausteine, von Werner Hilf und Anton Nausch, 1985, 400 S., kart., DM 69,-, Tewi-Verlag, München

Unter den Nachschlagewerken für den 68000-Prozessor ist der „Hilf-Nausch" sicherlich mit Abstand das beste und umfangreichste Buch. Wer ein Handbuch zu einem der modernen 16/32-Bit-Prozessoren schreiben möchte, muß ungewöhnlich akribisch arbeiten können, denn anders als in der 8-Bit-Ära kann heute keiner mehr die Abertausenden von Opcodes und Taktzyklen im Kopf behalten. Druckfehler und „Blackouts" sind damit praktisch vorprogrammiert. Um so erfreuter können wir bei dem vorliegenden Werk feststellen, daß die Zahl der typographischen und sachlichen Fehler sehr gering ist. Als Beispiel für einen „Blackout" können wir den ASL-LSL-Befehl zitieren. Zwar finden wir hier Querverweise („Vergleiche LSL-Befehl", „Vergleiche ASL-Befehl"), doch der Unterschied wird nicht korrekt herausgearbeitet. So heißt es beim LSL-Befehl: „Das V-Flag zeigt an, ob irgendwann während des Schiebens ein Vorzeichenwechsel aufgetreten ist" (Bd.1, S. 4-189). Genau dies ist jedoch falsch, denn sonst wären ASL und LSL identisch, wie übrigens auch L.Leventhal in seinem „68000 Assembly Language Programming" irrtümlicherweise behauptet hat.

Fachausdrücke der Informationsverarbeitung

Wörterbuch und Glossar, Englisch-Deutsch und Deutsch-Englisch, Redaktion K.Csikai, 1985, 1045 S. (Engl.-Dt.) und 642 S. (Dt.-Engl.), geb., DM 105,-, IBM Deutschland, Stuttgart (Vertrieb über Addison-Wesley Verlag, Bonn)

Dieses Wörterbuch ist nicht nur für Übersetzer geeignet, sondern auch für Wissenschaftler, die exakte Begriffsdefinitionen nachschlagen wollen, denn zu vielen Termini werden die offiziellen deutschen und amerikanischen Definitionen nach DIN usw. angegeben, wobei allerdings insbesondere IBM-Belange Berücksichtigung fanden. Es verwundert deshalb nicht, daß beispielsweise die Stichwörter „BASIC" und „COBOL" vorhanden sind, während „Pascal" und „C" fehlen. Überspitzt könnte man sagen: Was man nicht in den Normblättern findet, fehlt auch in diesem Wörterbuch. So wird etwa „Byte" verzeichnet, nicht jedoch „Halbbyte" oder „Nibble". Dies soll Sie jedoch nicht vom Kauf dieses Nachschlagewerks abhalten, das angesichts des im Verhältnis zum Seitenumfang ungewöhnlich niedrigen Preises eine überwältigende Fülle von Informationen bietet.

EPSLQ – Hardcopytreiber

Diskette und Begleitblatt, DM 98,-, Fa. Michael Gehret, Grönenbach

Die im Atari-ROM eingebaute Bildschirm-Dump-Routine funktioniert wegen der Zeilenvorschub-Esc-Sequenz nicht bei dem Epson LQ800. Die EPSLQ-Programme (u.a. als Desk-Accessory) beheben diesen Mangel. Allerdings erfolgt weder ein maßstabsgetreuer Ausdruck (1:1), was beim LQ800 möglich wäre, noch werden alle 24 Nadeln benutzt. Beim „Qualitätsdruck" wird sogar nur eine einzige Nadel angesteuert, wodurch der Ausdruck mehr als 5 Minuten dauert und zudem völlig verschmiert wirkt.

Atari ST – Tips und Tools zu C

von Olaf Hartwig, 1986, 215 S., geb., DM 48,-, Sybex-Verlag, Düsseldorf

Dieses Buch bietet dem fortgeschrittenen Atari-C-Programmierer eine Sammlung gerätespezifischer C-Routinen. Darüber hinaus finden Pascal- und BASIC-Programmierer Makros für spezielle Befehle dieser Programmiersprachen.

Atari ST – Arbeiten mit GEM

Band 1: Die AES-Bibliothek, von Gerd Sender, 1986, 320 S., geb., mit Diskette DM 68,-; Band 2: Die VDI-Bibliothek, von Holger Danielsson und Andreas Volkmann, 1986, 238 S., geb., mit Diskette DM 68,-, Sybex-Verlag, Düsseldorf

Beide Bände enthalten umfangreiche Beispielprogramme für den Umgang mit GEM, wobei das Schwergewicht naturgemäß auf der Programmiersprache C liegt. Damit der Leser nicht alle Programme abtippen muß, befinden sich die Quelltexte und fertig compilierten Programme auf den Begleitdisketten, die mit zusammen knapp 600K an Dateien viel Ware fürs Geld bieten.

Atari ST – Das Floppy-Arbeitsbuch

von Frank Aumann, Peter Maier und Ralf Stöpper, 1986, 166 S., geb., mit Diskette DM 69,-, Sybex-Verlag, Düsseldorf

Dieses ansprechend hergestellte Buch befaßt sich mit der Hardware der Atari-Laufwerke sowie mit der Programmierung in Assembler und C. Insbesondere wird auch darauf eingegangen, wie man den Floppy-Disk-Controller unter Umgehung der BIOS/XBIOS-Routinen direkt anspricht. Die mitgelieferte Diskette enthält einige Programme (überwiegend in ST Pascal plus geschrieben, allerdings ohne Quelltext), die sich sehen lassen können. Beispielsweise kopiert das Speedcopy-Programm eine 720K-Diskette in nur 70 Sekunden (ohne Formatierung), wobei beim 1040 ST die gesamte Quelldiskette auf einen Schlag eingelesen wird.

Apple ProDOS für Aufsteiger

Band 1, 2. Aufl., 203 S., DM 28,–
Band 2, 208 S., DM 30,–

von Ulrich Stiehl

Dr. Alfred Hüthig Verlag
Postf. 102869 · 6900 Heidelberg

Apple Assembler
Tips und Tricks

von Ulrich Stiehl
2. Aufl., 226 S., 3 Abb., kart., DM 34,–
ISBN 3-7785-1047-9

Dr. Alfred Hüthig Verlag
Postf. 102869 · 6900 Heidelberg

Das Simulationsprogramm Umweltdynamik

von Dieter Charchot

1. Eingangsbeispiel

Irgendwo gibt es ein kleines Waldstück, in dem die Natur noch intakt ist. Keine sterbenden Bäume, keine illegalen Müllkippen, keine Bodenverseuchung. Kräftige Pflanzen und eine gesunde Tierwelt sind hier noch anzutreffen.
In diesem kleinen Paradies leben etwa 500 Hasen, die von den 10 einheimischen Füchsen verständlicherweise sehr geschätzt werden.
Leider kann der Landwirt, dem dieses Fleckchen gehört, der Versuchung nicht widerstehen, auf der Hälfte des Areals einen lukrativen Campingplatz einzurichten.
Wie wirkt sich diese Halbierung des Lebensraumes auf unsere vierbeinigen Ureinwohner aus? Werden sich die beiden Tierarten im gleichen Verhältnis zueinander ebenfalls auf die halbe Anzahl reduzieren oder bewirkt ein derartiger Eingriff doch eine nachhaltigere Störung?
Tatsächlich zeigt eine Untersuchung, daß infolge der Halbierung der Weideflächen die Anzahl der Hasen drastisch zurückgeht. Die Nahrungsverknappung für die Füchse führt dazu, daß die Zahl dieser Tiere rapide absinkt und die Art innerhalb kurzer Frist völlig ausstirbt.

Untersuchungen dieser Art sind meist äußerst langwierig und kostspielig. Im vorliegenden Fall hat sich die nachhaltige Störung des natürlichen Gleichgewichtes als irreversibel erwiesen. Operation gelungen – Patient tot. Wieder einmal hat die Natur den kürzeren ziehen müssen.
Doch keine Aufregung. Die ganze Tragödie betraf weder putzige Langohren noch schlaue Füchse, sondern nur einige Elektronen und Schaltkreise im Apple IIe!
Es handelt sich um die Simulation einer Umweltsituation, eines von 30 Programmen auf der Diskette zum Buch „Umweltdynamik" von Hartmut Bossel, das im te-wi Verlag erschienen ist. Die Disketten zu diesem Buch sind übrigens nicht nur im Apple-Format erhältlich, sondern stehen für eine Reihe bekannter Rechner zur Verfügung.

2. Simulationsmodelle

Wer sich nun fragt, was ihn denn die Hasen und Füchse in irgendeinem Waldstück angehen, dem sei entgegnet, daß wir alle in einer gemeinsamen (gefährdeten) Umwelt leben, für die auch alle gemeinsam Verantwortung tragen sollten. Das setzt Kenntnis und Wissen um vielfältige Zusammenhänge voraus. Im übrigen ist diese Simulation nur ein (sogar recht einfaches) Beispiel für eine Räuber-Beute-Beziehung, die stellvertretend für eine ganze Reihe recht empfindlicher dynamischer Verkoppelungen von ökologischen Systemen steht.
Alle Simulationsmodelle sollen und können zudem als Anregungen dienen, selbst gestellte Probleme unterschiedlichster Art zu untersuchen. Die Fähigkeiten dazu erwirbt man bei der Lektüre des Buches, das eingehend Grundgedanken und Prinzipien der Computersimulation darstellt.
Welche Fortschritte die Computertechnik auf diesem Gebiet innerhalb kurzer Zeit vollzogen hat, geht auch aus dem Vorwort des Buches hervor, daß kein geringerer als Dennis Meadows geschrieben hat, Direktor des Resource Policy Center am Dartmouth College, USA, und Mitverfasser der bereits legendären Studie des Club of Rome „Grenzen des Wachstums".
Danach wurde diese Weltmodell-Untersuchung 1972 am Massachusetts Institute of Technology noch mit einem riesigen Aufwand an Technik und Mitarbeitern auf einem IBM-Großcomputer durchgeführt. Ähnliche Simulationen können heute bereits auf manchen tragbaren Kleincomputern laufen, oder eben auf dem guten, alten Apple.

3. Das Programm DYSYS

Das Vorgehen von Hartmut Bossel ist bestechend. Grundprogramm aller Demonstrationssimulationen ist das Programm DYSYS, das einen Rahmen darstellt, in dem die jeweiligen Einzeldaten zum lauffähigen Modell zusammengefügt werden.
Beim Apple geschieht das, indem zuerst DYSYS geladen wird und danach mit „EXEC Modellname" die Modelldaten aus den vorbereiteten Textfiles dazukommen. Übrigens ist die Anpassung an unterschiedliche Druckinterfaces oder Drucker kein Problem, da es sich bei DYSYS um ein übersichtliches BASIC-Programm handelt, in das die entsprechenden, individuellen Drucker-Interface-Kommandos leicht integrierbar sind.
Alle Modelle sind im Buch ausführlich erläutert, die Beziehungen der Einzelfaktoren zueinander sind teilweise mit aufwendigen Grafiken verdeutlicht. Änderungen der Formeln und Rechenoperanden werden auf diese Weise erleichtert.
Beim Start eines Modells werden bestimmte Anfangswerte abgefragt, mit denen die Berechnungen beginnen. Durch Veränderungen der Werte wird der Ablauf stark variiert.

4. Ausgabe der Ergebnisse

Die Ausgabe der Ergebnisse kann auf verschiedene Arten erfolgen, die Auswahl wird ebenfalls im kleinen Startmenü getroffen. Beeindruckend ist die grafische Ausgabe in Vergleichskurven: Beim Hase/Fuchs-Modell erhält man zwei Kurven (Anzahl Hasen und Anzahl Füchse) zusammen mit den laufend errechneten Zahlenwerten gleichzeitig mit der laufenden Simulation; danach können Einzelkurven (nur Hasen, nur Füchse) abgerufen werden, die im Maßstab beliebig veränderbar sind und so – bei entsprechender Auswahl – entweder direkt miteinander verglichen oder einzeln eingehender studiert werden können. Es folgt eine interessante dritte Auswertung, die die Abhängigkeit beider Kurven (Hasenkurve und Fuchskurve) voneinander grafisch darstellt. Diese Grafik ist zwar nicht sofort zahlenmäßig aussagekräftig, zeigt aber, ob das untersuchte System zur Stabilität oder zum Zerfall tendiert.
Alle Grafikausgaben werden erstaunlich schnell auf dem Bildschirm ausgegeben und können auf Knopfdruck auch auf einem Matrixdrucker dokumentiert werden. Danach kann mit den Momentanwerten weitergerechnet werden, es können Werte auch zuvor verändert oder das Modell ganz verlassen werden.
Besonders interessant sind Vergleichsläufe mit veränderten Rahmenbedingungen, bei denen die gedruckten Auswertungen dann im direkten Vergleich wichtige Auswirkungen auf einen Blick erkennen lassen.

5. Themengebiete

Dabei lassen sich mit den vorbereiteten Modellen die unterschiedlichsten Phänomene untersuchen. Ob nun die Ausbreitung einer Krankheit (oder eines Witzes), der Energie- und Rohstoffeinsatz unter unterschiedlichen Bedingungen, bestimmte Wachstums- und Zerfallsprozesse, Bevölkerungsdynamik unter Berücksichtigung verschiedener Altersklassen, Phänomene der Landwirtschaft, die Entwicklung einer bestimmten Tierart bei plötzlichem Wegfall der natürlichen Räuber, der Einfluß von saurem Regen auf einen Forstbestand – alles läßt sich simulieren, wenn man weiß, wie's gemacht wird.

6. Kybernetisches Wissen

Bisherige Schwierigkeiten mit dieser Art der Computernutzung sind nicht nur in rein programmtechnischen Gründen zu suchen. Voraussetzung hierfür ist eine Denkweise, die bisher nicht „geübt" wurde.
Die eingefahrenen Denkerfahrungen beschränken sich seltsamerweise nur auf ein Teilgebiet menschlichen Lebens. So klassifiziert der Autor in seiner Einleitung das vorwiegend vorhandene SACHWISSEN (Ein Pferd hat vier Beine, der Bürgermeister meiner Stadt heißt...), LOGISCHES WISSEN (Wenn ich Benzin anzünde, dann brennt es.) und NORMATIVES WISSEN (Lügen ist schlecht). Was er vermißt ist das KYBERNETISCHE WISSEN, die Kenntnisse über die Dynamik von Systemen. Daß hierbei ausgerechnet Beispiele aus der Umwelt benutzt werden, ist nicht ganz zufällig. Besonders ökologische Systeme sind normalerweise äußerst schwierig zu studieren, die Veränderung von Vorgaben ist nur begrenzt möglich oder wünschenswert (siehe Eingangsbeispiel).
Schon die notwendigen Zeiträume lassen effektive Untersuchungen

nur beschränkt zu. Dabei sind gerade derartige Studien unbedingt notwendig, um den Umgang mit derartigen dynamischen Systemen zu erlernen.

Bossel verwendet hier zur Verdeutlichung das Beispiel des Autofahrens. Diesen im wahrsten Sinn des Wortes dynamischen Vorgang kann man kaum theoretisch erlernen, man muß ihn üben, muß lernen, wie bestimmte Einflüsse auf das System wirken, und darauf richtig reagieren.
Ebenso notwendig ist diese Fähigkeit aber auch im Umgang mit unserer Umwelt, denn die Möglichkeiten des Menschen zu ökologischer Schädigung in globalen Maßstäben haben leider ebenfalls gewaltig zugenommen.

Um so wichtiger sind die „Fahrkurse" mit dynamischen Systemen, die z.B. die Kenntnisse um die empfindlichen Fließgleichgewichte in der Ökologie vermitteln und so vielleicht einige weitere Sünden verhindern könnten.
Die Erfahrungen, die bei der Beschäftigung mit dieser interessanten Materie gesammelt werden, können vielfältig genutzt werden. Das Buch und die Diskette bieten sich im schulischen Bereich ebenso an wie beim Studium. Betriebliche „Übungen" sind ebenso möglich wie wissenschaftliche Anwendungen, denn Systeme im hier angesprochenen Sinne gibt es mehr als genug.

Fazit

Die Beschäftigung mit diesem ungewöhnlichen Buch lohnt sich auf jeden Fall, der Preis von DM 59,- für das Buch und DM 29,80 für die fertige Diskette ist zwar hoch, aber gut angelegt. Viele Stunden Unterhaltung, Spannung und etliche „Aha"-Erlebnisse erwarten den Leser, der sich aufmacht, das DYSYS-Land zu erforschen. Gute Reise!

Literaturverwaltungsprogramm Bookends

von Dieter Charchot

1. Ausgangssituation

Jeder, der sich für ein bestimmtes Gebiet interessiert und mehr oder weniger viel darüber liest, kennt das: Ein bestimmter Artikel, ein Buchkapitel, eine Abhandlung wird dringend benötigt. Natürlich! Da hat man doch gerade vor kurzem im XY-Blatt in einer der letzten Ausgaben noch etwas darüber gelesen. Oder war es in der ABC-Zeitschrift oder in einem Fachbuch? Eine unangenehme Situation, die in den meisten Fällen zeitraubende Sucharbeit bedeutet und dann nicht immer von Erfolg gekrönt ist.

So manches Mal hat man sich in dieser Lage schon vorgenommen, Ordnung in den entsprechenden Unterlagen zu schaffen. Doch selbst dann ist das Ergebnis meist recht fragwürdig. Das Anlegen einer Quellenkartei und ihre Pflege, d.h. die ständige Aktualisierung der Daten, ist mit erheblichem Aufwand verbunden.
Sollte nicht auch dieses Problem mit einem Apple und einem Datenbankprogramm zu lösen sein?
Allerdings erfordert die Planung und Durchführung einer derartigen Literaturverwaltung bereits eine gewisse Erfahrung, um schon bei der Vorbereitung und Einrichtung alle Einzelheiten für einen langfristigen Betrieb zu berücksichtigen. Bestimmte Details können später die Gebrauchsfähigkeit des ganzen Systems in Frage stellen, manche wichtigen Funktionen können nur recht umständlich mit einem allgemein ausgelegten Datenbankprogramm verwirklicht werden. Für eng begrenzte Spezialanwendungen können diese Programme nicht immer optimal eingesetzt werden.

2. Abhilfe

Ein ausgesprochener Spezialist auf dem Gebiet der Quellenverwaltung ist das Programm Bookends (The Reference Management System) von Sensible Software, mit dem man endlich der Datenflut auf bequeme Art und Weise Herr werden kann.

2.1. Manual

Die neueste Programmversion Bookends Extended nutzt nun auch alle Möglichkeiten des Apple IIe (IIc) aus, erwartet also eine erweiterte 80-Zeichenkarte und damit 128K Gesamtspeicher.
Das Programm wird mit 2 Programmdisketten geliefert. Eine Programmdiskette ist also stets als Reserve verfügbar, ein Pluspunkt besonders für den kommerziellen Einsatz, bei dem es auf dauernde Verfügbarkeit ankommt.

Das spiralgeheftete, englischsprachige Manual beginnt mit einer Kurzanleitung für ganz „Eilige", danach werden alle Einzelheiten des Programms erläutert. Hinweise für die Installation von Bookends auf Harddisks und Verweise auf einige Utilities (z.B. wie über Modem empfangene Texte in Bookends-Format umgesetzt werden) runden das Informationsangebot ab. Am Ende des ausgezeichneten Handbuchs finden sich Anhänge mit Zusammenfassungen und ein ausführliches Stichwortverzeichnis.

2.2. Bedienung

Wird eine der Disketten gebootet, fragt Bookends zunächst nach dem aktuellen Datum (sofern keine Uhrenkarte im Computer steckt) und schaltet ins Hauptmenü, von dem aus man weitere Programmebenen anwählen kann. Neben diesen Auswahlmöglichkeiten findet man hier Initialisierungspunkte wie Drucker-Optionen oder Default-Werte, die man so wählen kann, daß sie bei späteren Programmstarts automatisch richtig eingestellt sind. Vom Hauptmenü aus können ProDOS-Funktionen wie Cat(alog), Delete, Init und Prefix (neusetzen) angewählt werden. Eine ausführliche Hilfsfunktion bietet Informationen zu den wichtigsten Bookends-Befehlen.
Die meisten Menüs und Funktionen kann man durch einfaches Drücken der ESC-Taste wieder verlassen, falls man von den angebotenen Optionen keinen Gebrauch machen möchte.
Bookends ist zwar ein amerikanisches Programm (Datumeingabe „Oct" statt „Okt"), aber mit den deutschen Umlauten gibt es erfreulicherweise keine Probleme.

2.3. Datensätze

Punkt 1 des Hauptmenüs führt in den Verwaltungsbereich. Hier wird zunächst das gewünschte Datenfile geladen, das die References (Datensätze) enthält, die einzelnen „Karteikarten".

Jeder dieser Einzeldatensätze enthält Angaben über Autor, Titel, Ort und Zeitpunkt der Veröffentlichung, darüber hinaus eine Liste von frei wählbaren Stichworten (Keywords) und Anmerkungen, z.B. über den Inhalt der Veröffentlichung.
Ist noch kein Datenfile vorhanden, werden neue Datensätze eingegeben. Dabei werden alle Kategorien nacheinander abgefragt. Vom vorher eingegebenen Datensatz können Eintragungen übernommen werden. Soll z.B. eine Artikelsammlung aus einer Fachzeitschift verwaltet werden, muß der Name der Zeitschrift nicht immer wieder neu eingetippt werden.

Die Editiermöglichkeiten sind ausreichend. Der Benutzer wird durch die Vorgabe von Wahlmöglichkeiten zuverlässig geführt. Die Zusammensetzung der Datensätze ist durchdacht und auf die späteren Verarbeitungsfunktionen abgestimmt. Besonders bemerkenswert ist, daß bei Bookends Extended *alle* Datensätze im Hauptspeicher stehen. Dadurch wird eine außerordentlich hohe Bearbeitungsgeschwindigkeit erreicht, und man kann unbesorgt mit den Daten arbeiten. Alle Änderungen werden erst nach ausdrücklichem Kommando auf der Diskette gespeichert. Das Programm erinnert an das Abspeichern, sofern durch eine bestimmte Menüanwahl die Gefahr besteht, daß die Daten im Hauptspeicher verlorengehen könnten.
Sollte eine Datensammlung umfangreicher werden, als es die Speicherkapazität verkraftet (ca. 160 - 320 Quellen), kann eine Verkettung der Teilfiles (Bücher, Bücher.2, Bücher.3 usw.) vorgenommen werden.

2.4. Suchfunktionen

In den „Karteikarten" kann beliebig vor- und zurückgeblättert oder gezielt gesucht werden. Unabhängig davon können komplette Listen der Autoren oder Schlüsselworte erstellt werden. Die Suchbegriffe können so sehr einfach auf ihre Schreibweise überprüft werden.
Die Suchfunktion kann auf bestimmte Datensätze beschränkt werden (z.B. Datensätze 3-145) oder alle Datensätze bearbeiten. Dabei können Begriffe einzelner Kategorien gesucht (Autoren, nur 1. Autor, Keywords, nur 1. Keyword) oder alle Einträge eines Datensatzes überprüft werden. Als „Joker" steht das „=" zur Verfügung.
Übersteigt die Datenmenge einer Datenbank die Kapazität des Hauptspeichers, fragt Bookends nach der Bearbeitung des ersten Teils automatisch, ob die Suche auch in den weiteren Teildatenbanken durchgeführt werden soll. Die Suchkriterien brauchen dann nicht mehr neu eingegeben zu werden.
Wird das Programm fündig, wird die Anzahl der „Treffer" gemeldet.

2.5. Ausgabeformat

Ist „Option off" gewählt, fragt Bookends, wie die Ausgabe erfolgen soll: Auf dem Bildschirm, auf dem Drucker, als Textfile auf Diskette oder als Anhang an ein bestehendes Textfile. Bei „Option on" erfolgt die Ausgabe automatisch auf dem Bildschirm.
Danach kann entschieden werden, ob die Daten im Eingabeformat ausgegeben werden oder ob ein neues Ausgabeformat verwendet werden soll. Die Formate können vom Benutzer frei definiert, abgespeichert und später je nach Bedarf aufgerufen werden. Eine Reihe dieser Formate sind auf der Masterdiskette bereits vorbereitet, ebenso einige Beispieldateien mit Datensätzen, so daß sich jeder neue Anwender anhand dieses „Experimentiermaterials" und des ausgezeichneten Manuals schnell und einfach mit Bookends vertraut machen kann.

Fazit

Das Programm ist professionell gemacht. Das schlägt sich allerdings auch im Preis nieder. Für Bookends Extended muß der ordnungsliebende Apple-User ca. DM 580,–, für Bookends ca. DM 480,– anlegen.
Das Programm bewegt sich damit allerdings immer noch in Größenordnungen, die für moderne Software an der Tagesordnung sind (s. ProDOS-Applewriter, Appleworks usw.).
Es handelt sich hier eben um einen ausgesprochenen Spezialisten, der sich seine besonderen Fähigkeiten entsprechend honorieren läßt.

MICOL-BASIC-Sonderangebot

Compiler 2.0 (Diskette und Handbuch) . . . . . DM 138,–
(ab Lager lieferbar, solange Vorrat reicht)

GFABASIC-Sonderangebot

Interpreter 2.0 (Diskette und Handbuch) . . . . . DM 138,–
Compiler 1.7 (Diskette und Handbuch) . . . . . DM 138,–
(ab Lager lieferbar, solange Vorrat reicht)

Hüthig Software Service · Postfach 10 28 69 · 6900 Heidelberg

Schreiben mit der Maus

Textverarbeitungsprogramm Multiscribe

getestet von Reinhard Frank

Keine Frage, die Maus hat zum Sturm auf die Apple-II-Besitzer geblasen, um einen Hauch von Macintosh auf das Desktop zu zaubern. Ob in Pascal, UCSD oder Kyan, bei Grafiksoftware oder gar BASIC, die Maus ist immer dabei.
Seit einiger Zeit gibt es auch Textverarbeitungsprogramme, deren Funktionen mit der Maus gesteuert werden können. Mousewrite (Roger Wagner Publishing, im Peeker, 7/86, besprochen), Mouseword (International Solution) und Multiscribe (Style Ware) sind drei davon. Sie sind amerikanischer Herkunft mit englischen Handbüchern, was einem halbwegs versierten „Peeker" wohl keine Schwierigkeiten macht. Aber es gibt noch andere „Schnittstellenprobleme", z.B. Ä, Ö und Ü und das gute deutsche ß.
Das Programm meiner Wahl war Multiscribe. Und dort gibt es weder Umlaute noch ß. Um das Fazit vorwegzunehmen: Ein typisches Textverarbeitungsprogramm ist es nicht, und auf die Maus verzichte ich meistens, wenn ich damit arbeite. Trotzdem arbeite ich recht häufig mit Multiscribe. Wie ist das möglich?

1. Handbuch

Multiscribe kommt mit einer doppelseitigen Diskette und einem furchteinflößenden Handbuch von 250 Seiten ins Haus. Man liest es am besten – wie viele amerikanische Handbücher – von hinten nach vorne. Vorne wird erklärt, wie man erst die Diskette einlegt oder was eine Diskette überhaupt ist, welche Seite nach oben zeigen soll, wie man den Computer einschaltet usw. Hinten ist die „Reference" angefügt, die auf einer Handvoll Seiten alles Wesentliche zusammenfaßt. Aber auch die „Reference" braucht man schon nach kurzer Zeit nicht mehr, weil alles sehr einfach zu bedienen ist.
Multiscribe zeigt nach dem Booten eine obere Menüleiste mit „Edit", „Format", „File" u.ä., von wo aus in Pulldown-Manier nützliche Funktionen zugänglich sind. Das macht man natürlich mit der Maus, und das ist auch sehr schön. Aber es geht auch ohne Maus und nach meiner Erfahrung sogar besser, rascher und genauer von der Tastatur aus. Die meisten Tastenbefehle sind auch in den Pulldown-Menüs zu sehen. Diese wiederum sieht man nach Betätigen der ESC-Taste. Mit den Pfeiltasten ist man rasch an der gewünschten Stelle.
Was ist aber mit den Ä, Ü und dem ß? Nach einer kurzen Textprobe sieht man den Salat: Nur eckige und geschweifte Klammern sowie die Tilde anstelle des Eszett.

2. Schriftarten und -schnitte

Nun steht in der Menüleiste auch „Font" (= Schriftart), beim Pulldown erschrickt man vor zwölf respektablen Namen wie „Hemingway", „Mark Twain" und „Shakespeare". Sie entpuppen sich beim Anwählen als verschiedene Schriftarten. Jeder Font kann mit „Style" (= Schriftschnitt) zusätzlich in sechs Varianten erscheinen, von „Plain" = Grundschrift bis „Shadow" = Konturenschrift. Jetzt wird auch klar, warum Multiscribe kein „ß" erzeugen kann: Der ganze Text, in welcher Schriftart und welchem Schriftschnitt auch immer, wird auf dem Grafikschirm dargestellt. Also muß jeder Buchstabe als kleine Grafik definiert sein. Der deutsche Zeichensatzgenerator in meinen Apple wird nicht aktiviert.
Zur Beruhigung: Multiscribe kann auch das ß. Man muß es ihm nur beibringen. Der für mich inzwischen wichtigste Teil von Multiscribe ist „Multifont", ein außerordentlich rascher, eleganter und leicht bedienbarer Editor für Schriften. Ich habe mir die Programmteile von Multiscribe, die übrigens frei kopierbar sind, so auf zwei Disketten verteilt, daß ich rasch vom Textschreiben auf Schriftzeichenentwicklung umschalten kann. Nun kann ich Multiscribe auf besondere Weise nutzen, denn ein vollwertiges Werkzeug zur Textbearbeitung ist es nicht. Auch mir als ungeübtem Tipper und mäßigem Briefeschreiber ist es einfach zu langsam. Zwei oder drei Seiten Text bringt man noch gut in den Speicher, aber dann muß man Blickkontakt zum Bildschirm halten, um zu sehen, ob auch alles angekommen ist, was man schreiben wollte. Die grafische Darstellung hat eben ihren Preis.
Ich beschäftige mich mit Chemie und bisher hat mir kein Textprogramm chemische Formeln, Flußdiagramme oder Versuchsapparaturen mitten im Text unterbringen können. Mit Multiscribe ist dies möglich. Der Buchstabe „R" meines selbsterstellten Fonts „Chem. apparat" enthält ein Reagenzglas, „K" einen Kolben, „G" für „GIFT" sogar einen Totenkopf. Die Zahlen 1 bis 9 enthalten Leerzeichen und Striche mit der Breite 1 bis 9, so daß jedes Symbol pixelgenau positioniert werden kann. Die Zeichenbreite ist für jeden Buchstaben zwischen 1 und 28 (!) einstellbar, die Zeichenhöhe für einen Zeichensatz zwischen 4 und 28 (!). Außerdem kann man den Zeilenabstand 0 wählen, so daß zwei oder mehrere Zeilen lückenlos zusammengeklebt werden können (wenn der Drucker mitmacht).

3. Ausdruck

Soll der so gestaltete Text nun im Grafikmodus ausgedruckt werden, muß man zuvor die Druckerschnittstelle und den Drucker festlegen, was bei meinem Apple IIc und dem Imagewriter keine Sorgen bereitet. Die Liste weiterer Schnittstellen ist mit ca. 40 recht umfangreich, doch können keine eigenen Druckertreiber installiert werden. Dann muß gepatcht werden, was nicht ganz einfach sein dürfte.
(Anm.d.Red.: Die Grafiktreiber finden sich leider verstreut im Hauptmodul. Eigene Treiber müßten deshalb sehr kurz und außerdem relokativ sein, da Multiscribe mit Overlay arbeitet und sich deshalb nicht ohne weiteres ermitteln läßt, in welchen Bereich der eigene Grafiktreiber geladen wird. Um dies zu untersuchen, wurde uns von Softline in Oberkirch das Programm Multiscribe kurzfristig ausgeliehen. Von dieser Firma ist übrigens inzwischen eine erweiterte Version erhältlich.)

Zusammenfassung: Kein Programm, um rasch ein schriftliches Dokument zu erstellen, aber ein gutes und auch preiswertes Programm, wenn man auf Sonderzeichen Wert legt.

Schriftarten

Macroworks –

Eine Erweiterung für Appleworks

von Franz-Josef Hüskens

Glaubt man der Werbung, so ist Appleworks das meistbenutzte integrierte Softwarepaket für den Apple IIe/c. Dem Anwender bietet es Textverarbeitung, Datenverwaltung und Tabellenkalkulation. Viele der Befehle des Programmpakets können wahlweise mit der „offenen" (oA) oder mit der „geschlossenen" Apfeltaste (gA) ausgelöst werden. Die Softwarefirma Beagle Brothers hat entdeckt, daß Appleworks einen bestimmten Speicherbereich frei läßt. Das „Hilfs"-Programm Macroworks nutzt diesen Bereich.

Macrodefinitionen

Macroworks ist ein Programmpaket, das zusammen mit Appleworks benutzt werden muß. Es stellt dem Appleworks-Anwender Macros zur Verfügung. Macros sind zu *einem* Tastendruck zusammengefaßte Befehls- oder Textsequenzen, die normalerweise mehrere Tastendrücke umfassen. So kann man z.B. für die Textverarbeitung ein Macro vereinbaren, das Name und Anschrift in einen Text einfügt oder einen Absatzeinschub von drei Zeichen Länge vornimmt. Für die Appleworks-Datenbank kann man z.B. ein Macro vereinbaren, das alphabetisch sortiert, für das Rechenblatt ein Macro, das den Cursor in das Feld A1 bringt usw.

Die Macros und – falls gewünscht – auch Kommentare zu diesen werden mit der Textverarbeitung von Appleworks erstellt. Dafür stehen 4095 Bytes zur Verfügung. Jede Taste der Tastatur kann mit einem Macro belegt werden, das Vereinbaren von Macros für Control-Zeichen ist ebenfalls möglich. Ausgenommem sind nur die Del-Taste und die Ziffern 0-9. Die Del-Taste ist mit einer Löschfunktion fest belegt, die Tasten 0-9 können vorübergehend mit Macros belegt werden, und zwar während man mit Appleworks arbeitet. Pro Zifferntaste kann ein 70 Zeichen langes Macro gespeichert werden. Diese „zeitweiligen" (temporären) Macros können so lange verwendet werden, bis man neu bootet oder das Macro umdefiniert. Zusätzlich haben sie noch den Vorteil, daß man mit ihnen neue Macros bei der Erstellung testen kann, bevor man diese in einer Datei ablegt.

Für bestimmte Tasten wie „Esc", „Ctrl", „Del", „Return" usw. werden bei der Programmierung „Tokens" (Schlüsselwörter) verwendet, die in spitzen Klammern angegeben werden. Macros können andere Macros oder auch sich selbst aufrufen. Nach Abschluß der Programmierung wird der Macro-File gespeichert und dann mit einem speziellen Compiler bearbeitet. Danach wird der File MACRO.SYSTEM auf der Appleworks-Startdiskette abgelegt und der File APLWORKS.SYSTEM in APLWORKS.SYS umbenannt. Damit ist aus der Appleworks-Startup-Diskette eine sogenannte „Macro-Master-Diskette" geworden. Bei jedem Booten wird nun der File MACRO.SYSTEM mit den vereinbarten Macros geladen.

Einschränkungen

Nachteilig bei der Benutzung vom Macroworks ist, daß die Macros, die für *einen* Appleworks-Teil (z.B. Textverarbeitung) geschrieben wurden, bei einer anderen Anwendung (z.B. Datenbank) nicht richtig funktionieren. Allerdings wird der Apple bei der Benutzung eines „falschen" Macros nur piepsen und vielleicht bestimmte Befehlseingaben auf den Bildschirm schreiben. Das Handbuch verschweigt weder diesen noch andere Mängel und empfiehlt in diesem speziellen Fall, für jeden Anwendungsbereich verschiedene Startup-Disketten mit den entsprechenden MACRO.SYSTEM-Files anzufertigen.

Anpassung

Appleworks existiert in verschiedenen nationalen Versionen. Zur Anpassung von Macroworks an die europäischen Belange befindet sich auf der Diskette der Macroworks-Version 2.3 ein File mit Namen EUROPE bzw. bei der Version 2.5 das Programm FOREIGN. Damit kann man Macroworks an die geänderten Funktionen der französischen, britischen und deutschen Appleworks-Versionen anpassen. Laut Handbuch arbeitet Macroworks mit den Appleworks-Versionen 1.0, 1.1, 1.2 und 1.3. Getestet habe ich das Programm mit Appleworks 1.2 und 1.3; damit lief es ohne Probleme.

Macroworks-Beigaben

Wer Beagle-Brothers-Software kennt, weiß, daß neben dem „Hauptprogramm" eines Pakets oft weitere nützliche Programme und Tips und Tricks mitgeliefert werden. Damit kann – allerdings nur das amerikanische – Appleworks in gewissen Funktionen (Fehlerhinweiston, Hilfsseiten) zusätzlich verändert werden.

Neben dem Macroworks-Programmsystem bietet das Paket noch folgende nützliche Programme:

ALPHA-CAT fertigt einen zweispaltigen und auf Wunsch alphabetisch sortierten Ausdruck eines ProDOS-Disketteninhaltsverzeichnisses an. Wahlweise können alle Dateien der Diskette oder nur die Appleworks-Dateien aufgenommen werden.

ANALYST analysiert die Dateien der Appleworks-Textverarbeitung. Damit unterstützt es den Anwender bei der Analyse seiner Texte und hilft bei der Erstellung von Registern. Dazu zählt ANALYST alle Wörter der Datei und gibt diese – auf Wunsch sogar alphabetisch sortiert – aus. Abgeschlossen wird die Liste mit der Wortanzahl, der Anzahl gleicher Wörter, der Anzahl der Buchstaben bzw. Zeichen und der durchschnittlichen Anzahl der Buchstaben je Wort. ANALYST arbeitet nur mit Dateien bis 15K (30 Blocks) Größe.

GALLEY ist ein sehr nützliches Programm. Es druckt einen entsprechend vorbereiteten Textfile zwei- oder dreispaltig aus. Für einen einwandfreien Ausdruck müssen der linke und rechte Rand des Schriftstücks bestimmte Werte haben, die im Handbuch vermerkt sind. Außerdem muß zum „Erzeugen" der zweiten (bzw. dritten) Spalte mit oA-PSV ein Seitenvorschub „künstlich" eingegeben werden. GALLEY arbeitet mit Dateien, die nicht größer als 20K (40 Blocks) sind; größere Schriftstücke müssen in mehrere kleine aufgeteilt werden.

Zusätzlich enthält die Diskette ein verbessertes BYE-Kommando. Hierbei werden alle „.SYS"-Files und alle Subdirectories der Diskette in Laufwerk 1 angezeigt. Mit den Pfeiltasten kann der gewünschte Systemfile ausgewählt und mit „Return" gestartet werden. Oder es werden alle „.SYS"-Files eines ausgewählten Subdirectories angezeigt. Will man ein anderes Laufwerk (auch RAM-Disks) benutzen, so kann man es mit „ESC" anmelden.

Auf der Diskette befinden sich auch Files, die verschiedene Apple- bzw. Appleworks-Erweiterungen wie die RAM-Karten von CHECKMATE und Applied Engineering unterstützen. Die Flipper-Karte von Cirtech (s. Bericht im Peeker, Heft 10/86) wird von Macroworks nicht unterstützt, arbeitet jedoch einwandfrei mit Appleworks 1.3, da diese Karte die original Apple-Speichererweiterung „simuliert". Im Betrieb mit Macroworks, Appleworks 1.3 und der Flipperkarte gab es keine Probleme.

Fazit

Das Handbuch in gewohnter „Beagle-Brothers-Qualität" bietet neben ausführlichen Programmbeschreibungen noch 6 Seiten Tips zu Appleworks und Macroworks.

Macroworks ist trotz den genannten Einschränkungen ein Programm, das man jedem Appleworks-Benutzer empfehlen kann. Der Preis von ca. DM 140,– scheint mir gerechtfertigt.

Die Soundkarte ADD 6

getestet von Hans-Martin Eng

Eine der bekanntesten Sound-Interfacekarten für den Apple II dürfte wohl das Mockingboard sein. Es fällt vor allem durch seinen hohen Preis auf (ca. DM 600,-), wird andererseits von vielen Softwareprodukten (z.B. Music Construction Set) angesteuert. Die ADD 6 ist nun eine Eigenentwicklung der Firma Tombstone-Micro. Sie zeichnet sich gegenüber dem Mockingboard vor allem durch einen niedrigeren Preis aus.

1. Lieferumfang

Für seine DM 350,- bekommt man außer der Soundkarte noch eine Diskette mit Demoprogrammen und Ansteuerungssoftware und eine 21seitige Loseblattsammlung namens Handbuch. Ein Anschlußkabel für die Heim-Stereoanlage fehlt leider, hier ist der Käufer zum Selbstlöten aufgerufen oder muß sich aus dem Adapterangebot des HiFi-Handels bedienen.

2. Hardware

Auf der 15,1 x 7,1 cm großen Grundfläche der Platine sind vier Soundchips vom Typ SAA 1099 von Valvo als aktive Elemente untergebracht, die Ansteuerung und Regelung besorgen zwei Steuer-ICs, zwei Quarze, zwei Trimmkondensatoren, 8 Trimmpotentiometer und diverse Widerstände und Kondensatoren. Das Ausgangssignal der ADD 6 wird über die Trimmpotentiometer auf eine 6,3 mm Stereoklinkenbuchse herausgeführt.
Leider unterließen die Konstrukteure eine Verstärkung des Signals auf der Platine. So liefert die Karte noch nicht einmal den HiFi-üblichen Ausgangspegel (d.h. man muß den Verstärker weiter aufdrehen als bei normalen HiFi-Geräten). Obwohl es der Anblick der Klinkenbuchse nahelegt – einen Kopfhörer braucht man auch nicht anzuschließen. Dafür reicht der Ausgangspegel schon gar nicht aus.
Auch die Lage der Klinkenbuchse auf der ADD 6 gefällt mir nicht ganz. Dadurch, daß sie ganz links liegt, muß der Klinkenstecker so eingesteckt werden, daß das Kabel in den Apple hineinweist. Um dann wieder aus dem Gehäuse herauszukommen, blieb mir nichts anderes übrig, als das Kabel einmal um die ADD 6 herumzuführen. Diese Anordnung ist mir um so unverständlicher, als auf der Platine genügend Platz vorhanden ist, den Stecker weiter nach innen zu legen. Würde dies geschehen, könnte das Anschlußkabel auf dem kürzesten Wege das Gehäuse des Computers verlassen. Eine bessere Lösung wäre allemal, das Kabel direkt an den Ausgang der Karte zu löten und (am Ende mit Cinch- oder DIN-Steckern versehen) aus dem Rechner herauszuführen. Dann entfiele zumindest für eine Gruppe von Käufern das Problem des Adapterkaufs.

Durch die Verwendung des Soundchips SAA 1099 von Valvo ist die ADD 6 voll zum Mockingboard *in*kompatibel. Was soll's, zumindest theoretisch sind diese Soundchips dem Mockingboard-Chip überlegen. Dem Besitzer der ADD 6 stehen nämlich zur Lautuntermalung seiner Programme insgesamt 24 Rechteckgeneratoren, 8 Hüllkurven- und 8 Rauschgeneratoren zur Verfügung. Der Frequenzbereich jedes Rechteckgenerators reicht über acht Oktaven hinweg von 31 Hz bis 7,81 kHz. Die Amplitude jedes Rechteckgenerators kann zwischen 0 und 15 festgelegt werden. Damit ergeben sich vielfältige Klangmöglichkeiten. Allein die Bezeichnung Synthesizer empfinde ich in diesem Zusammenhang denn doch etwas übertrieben.
Leider verfügt die ADD 6 ebenso wie das Mockingboard weder über eigenes ROM noch eigenes RAM. So muß die Treibersoftware in den Hauptspeicher des Apple geladen werden. Dies ist vielleicht kein großes Manko, doch legt die geringe Packungsdichte der Bauelemente die Einbeziehung weiterer Funktionen nahe.

3. Installation

Die Installation der ADD 6 im Apple ist unproblematisch. Der Anschluß an die Stereoanlage (oder einen anderen Verstärker) setzt hingegen technischen Sachverstand voraus. Am einfachsten dürfte hier noch der Griff zum Lötkolben sein.

4. Die Software

Auf der Diskette befinden sich im wesentlichen zwei Programme. Zum einen der sogen. „SPICE" (SPICE ist ein Akronym für Sound Programming Interactive Computer Environment), zum anderen der SSP-Menue-Master, ein menügesteuertes Lernprogramm, mit dem man die einzelnen Register der Soundchips direkt ansteuern kann. Dagegen fehlt ein Programm, mit dem man nach Einstellen des grundsätzlichen Klangs direkt Noten eingeben kann (etwa wie das Music Construction Set). So erscheint es mir unmöglich, mit SPICE allein ein längeres Musikstück zu programmieren. Mit Assemblersprache geht es ein wenig einfacher.

4.1. SPICE

Der SPICE-Interpreter ist eine Art maschinennahe Sprache, die die Ansteuerung der einzelnen Register der Soundchips erleichtert. Auf der Programmdiskette befindet sich zusätzlich ein in Applesoft BASIC geschriebener SPICE-Editor, mit dem man SPICE-Programme editieren, auf Diskette abspeichern und von Diskette einlesen kann.
Der SPICE-Interpreter selbst ist hinreichend schnell, der Editor jedoch unzumutbar langsam. Zum Beispiel kann das Auflisten eines ca. 20zeiligen Programms bis zu sechs Sekunden dauern. Was mir ebenso unangenehm auffiel, ist die Tatsache, daß eine Kommentierung der SPICE-Programme nicht möglich ist. Zudem fehlt die Möglichkeit, SPICE-Listings als Textfiles auf Diskette abzuspeichern.
Zu bemängeln ist ferner, daß ein Unterprogrammaufruf weder von SPICE-Programmen noch von Maschinenprogrammen möglich ist. Dies sollte man doch zumindest dem Hobby-Programmierer mit auf den Weg geben, weil sonst ein uneffektiver Spaghetti-Code entsteht. Ohne weiteres ist jedoch der Aufruf von SPICE-Routinen von Assemblerprogrammen aus möglich.

4.2. SSP-Menue-Master

Der SSP-Menue-Master ist ein sehr hilfreiches Lernprogramm. Mit diesem kann man ohne große Umstände beliebige Klangbilder entwerfen und austesten. Leider fehlt hier die Möglichkeit, die Registereinstellungen auf Diskette zu konservieren, so daß man gezwungen ist, diese von Hand abzuschreiben.

5. Das Handbuch

Die 24seitige Dokumentation ist in einem Schnellhefter untergebracht. Das erste Kapitel besteht aus einer kurzen Einführung in die ADD 6. Das zweite Kapitel stellt den verwendeten Soundchip SAA 1099 vor, Kapitel 3 befaßt sich mit dem SPICE-Interpreter, und Kapitel 4 schließlich führt in die Benutzung des SSP-Menue-Masters ein.
Das Handbuch wurde auf einem Matrixdrucker im NLQ-Modus erstellt. Dagegen ist nichts einzuwenden. Die orthographischen Unzulänglichkeiten störten mich schon eher. Alle notwendigen Informationen werden jedoch dem Leser zur Verfügung gestellt.

6. Demo-Programme

Also, ehrlich gesagt, die Demoprogramme sind eher ein Armutszeugnis. Sie sind alle in SPICE geschrieben, nur ein einziges jedoch wurde seinem Namen auch wirklich gerecht, nämlich „TRAINS" – man glaubt wirklich, eine Dampflokomotive zu hören. Das Nebelhorn war eher eine Enttäuschung, die Sirene alarmierte mich nicht im geringsten. Man kann dies jedoch entschuldigen, da die ADD 6 wirklich ein brandneues Produkt ist (unser Testexemplar trägt die Seriennummer #3).

7. Fazit

Ein abschließendes Urteil zu finden, fällt mir offen gestanden recht schwer. Kleine Mängel am einen oder anderen Ende trüben den hinterlassenen Gesamteindruck. Speziell stören mich der zu niedrige Ausgangspegel der ADD 6, die nicht vorhandene Softwareunterstützung durch andere Softwarehäuser, die fehlende Möglichkeit, Unterprogramme von SPICE aus aufzurufen und die Tippfehler im Handbuch. Positiv hervorzuheben ist vor allem der niedrige Preis. Guten Gewissens empfehlen könnte ich die ADD 6, verfügte sie über eine erste Verstärkerstufe, einen Cinch- oder DIN-Anschluß an die HiFi-Anlage und einen Noten-Editor nach Art des Music Construction Set. Die restlichen Mängel kann der erfahrene Anwender selbst ausbügeln.
Redaktioneller Nachtrag: Auf unsere Nachfrage stellte die Fa. Tombstone-Micro in Aussicht, daß künftig die ADD 6 mit einem Adapterkabel Klinke-DIN ausgeliefert wird. SPICE- und Assemblerunterprogrammaufrufe von SPICE aus sollen in die nächste Version von SPICE aufgenommen werden, ein Programm zur interaktiven Eingabe von Musik und ein SPICE-Assembler seien in Vorbereitung.

Datenverarbeitung mit dem Sinclair QL
von R. Fischer
1986, 198 S., zahlreiche Abb., DM 42,–
Dr. Alfred Hüthig Verlag, Heidelberg
Gliederung
Von (Super-)BASIC zu QL-ARCHIVE – Der Umgang mit QL-ARCHIVE – Anlegen einer QL-ARCHIVE-Datei – Arbeiten mit Masken – Arbeiten mit mehreren Dateien – Datenübertragung von und zu anderen Programmiersystemen – Weitere Beispiele – Anhangskapitel: Einrichten von QL-Programmen, QL-ARCHIVE-Handhabung, Anschluß von Druckern, Bibliothek für QL-ARCHIVE – Zuordnung deutscher und englischer Schlüsselwörter, Dokumentation der verwendeten Programme – Literaturverzeichnis – Register
Bemerkungen
Diese Buch ist eine hervorragende Ergänzung zum Handbuch des Sinclair QL. Es enthält u.a. ein Übersetzungsprogramm in SuperBASIC und tabellarische Gegenüberstellungen der englischen und deutschen Version dieses Mikrocomputers. Angesprochen sind v.a. Leser, die nicht nur marktgängige Software benutzen, sondern selbst Problemlösungen programmieren wollen. Dazu werden Beispiele aus Hobby, Schule und Beruf angeführt. Im Mittelpunkt des Buches steht das Erstellen und Bearbeiten von Dateien. Dem Leser werden die Möglichkeiten von SuperBASIC und die Handhabung der Dateiverwaltungssprache QL-ARCHIVE erläutert. Die Integration in andere QL-Programme wie Textverarbeitung oder Tabellenkalkulation und Grafik wird demonstriert. Die umfassenden Informationen über Programmiermethoden und die Hinweise zur Erstellung stukturierter Programme machen dieses Buch für Anfänger und Fortgeschrittene geeignet.

Sinclair QL-Anwenderhandbuch
Tips. und Beispiele in BASIC und Maschinensprache
von D. Kiesenberg
1984, 121 S., kart., DM 39,80
Dr. Alfred Hüthig Verlag, Heidelberg
Gliederung
Vorbemerkungen – Die Tastatur des QL – Die Mikro-Drive-Taste – Bedienung der mitgelieferten Programme – ABACUS: das elektronische Arbeitsblatt (Kalkulator) – ARCHIVE: Das Datenbankprogramm (Archivar) – EASEL: Das Grafikprogramm (Grafiker) – QUILL: Der Textprozessor (Sekretär) – Die SuperBASIC-Befehle – Das Betriebssystem QDOS, der Speicheraufbau des QL – Die QL-Anschlußbelegungen – Programmierung des 68008 – Tips & Tricks – Indexregister
Bemerkungen
Das Sinclair-QL-Anwenderhandbuch liefert dem Benutzer eine Vielzahl nützlicher Informationen für das Arbeiten mit dem QL. In der Einführung werden dem Neuling wichtige Computer-Fachbegriffe und die verschiedenen Anschlußmöglichkeiten des QL erklärt. Die Funktion und Bedienung der mitgelieferten Programme ABACUS, ARCHIVE, EASEL und QUILL wird detailliert erläutert; am Ende jedes Abschnitts werden die programmeigenen Befehle in einer Übersicht aufgelistet. Durch ein Anwendungsbeispiel wird die Arbeitsweise des „elektronischen Arbeitsblattes" ABACUS verdeutlicht. Im anschließenden Kapitel erhält der Leser eine Einführung in SuperBASIC: Alle Befehle von SuperBASIC werden beschrieben und in einer alphabetisch geordneten Übersicht zusammengefaßt. Danach werden das QL-Betriebssystem QDOS und die QL-Speicherbelegung behandelt. Die QL-Anschlußbelegungen werden in einem eigenen Kapitel mit Abbildungen und Erläuterungen der Signalnamen dargestellt. Im nächsten Kapitel beschäftigt sich der Autor mit der Programmierung des Prozessors 68008 von Motorola. An dieser Stelle kann natürlich keine vollständige Einführung in die Assemblerprogrammierung des 68008 erfolgen, aber der Leser erhält eine Übersicht über alle 68008-Befehle und damit die Möglichkeit, mit dem QL Maschinenprogramme zu schreiben. Als kleinen Leckerbissen bekommt der Leser unter dem Stichwort „Tips und Tricks" zum Abschluß noch einige kleine Utility-Programme geliefert.

Computer-Grafik: Einführung – Algorithmen – Programmentwicklung
Ein Arbeitsbuch mit zahlreichen Programmen und Hardware-Applikationen für Plotter und Bildschirm
von J. Plate
1987, 415 S., 252 Abb., kart., DM 68,–
Franzis Verlag, München
Gliederung
Einführung – Grafische Ausgabegeräte – Grafische Eingabegeräte – Linien zeichnen – Kreise und Ellipsen – Koordinatensystem – Darstellung von Funktionen und Meßwerten – Schildkrötengrafik – Schraffieren und Flächen füllen – Ein Grafikpaket – Der Selbstbauplotter – Grundlagen dreidimensionaler Grafik – Hidden-Line und Hidden-Surface – Zweidimensionale Flächen – Bewegte Grafik – Grafikdateien – CAD – Bildabtastung und -digitalisierung – Weiterführende Techniken – Zeit verschwenden mit dem Plotter – Literatur – Register
Bemerkungen
Der Anwender dieses Arbeitsbuches findet eine gründliche und praxisorientierte Anleitung zum Erstellen, Bearbeiten und Auswerten von Computergrafiken. Dabei wird das Thema schrittweise angegangen: von der ersten Linie über Koordinatensysteme, Balkendiagramme bis zu zwei- und dreidimensionalen Grafiken und höheren Techniken wie dem Schraffieren oder CAD-Lösungen.

Der Autor beschränkt sich in diesem Band nicht auf Programme zur Bildschirmdarstellung von Computergrafiken, sondern orientiert sich stärker am Einsatz von Plottern und die entsprechend gestalteten Programmen. Plotter sind heute schon relativ günstig zu haben; der Selbstbauplotter, der im Buch beschrieben ist, macht auch weniger finanzstarken Interessenten diesen neuen Bereich zugänglich. Neben der Bauanleitung für den Plotter werden zahlreiche Schaltungsbeispiele sowie Beschreibungen der Grafikperipherie und deren Ansteuerung behandelt. Die zahlreichen Programmbeispiele wurden auf dem Apple II und dem IBM-PC erarbeitet, die Verfahren und Algorithmen lassen sich jedoch größtenteils auch auf andere Rechner übertragen. Wegen der Übersichtlichkeit und Portabilität wurden die Programme meist in TURBO-Pascal erstellt, einige Beispiele sind in BASIC oder Assembler geschrieben. Computerspezifische Teile wurden soweit als möglich vermieden, um ein Anpassen an andere Rechner zu erleichtern. Bei dem vom Autor verwendeten Plotter handelt es sich um ein MP 1000 von Graphtec (Watanabe), es können jedoch auch andere Plotter eingesetzt werden. In diesem Buch findet der interessierte Leser eine Menge Information für seine erfolgreiche Grafik-Arbeit am Computer.

Die Programmiersprache Lisp
Eine Einführung in die Sprache der Künstlichen Intelligenz
von G. Schoffa
1987, 184 S., kart., DM 48,–
Franzis Verlag, München
Gliederung
Allgemeines über Lisp – Die ersten Versuche in Lisp – Die Datenstruktur in Lisp – Die Primitiven zur Erkennung des Datentyps – Zuweisung von Werten an Symbole – Vergleichen der Objekte – Zugriff auf Elemente von Listen – Nichtdestruktive Veränderung der Listen – Destruktive Veränderung der Listen – Die logischen Funktionen – Programmieren in Lisp – Programmsteuerung – Ein- und Ausgabe – Programmierbeispiele – Diverses – Glossar – Anhang – Literatur – Register
Bemerkungen
Auf der Basis des Lisp-Dialektes Common-muLisp-86 führt der Autor auf gehobenem Niveau in die Benutzung der Programmiersprache Lisp ein, der nach FORTRAN ältesten höheren Computersprache. Der verwendete Dialekt wird besonders im PC-Bereich benutzt, gehört aber nach einer Entscheidung der IJCAI-Konferenz von Los Angeles zum inoffiziellen Standard. Wegen der Komplexität der Sprache und der Programmiertechniken erhält der Leser eine stufenweise und praxisorientierte Einführung in die Interpretersprache Lisp. Dabei wird er systematisch mit allen wichtigen Lisp-spezifischen Strukturen und der Denkweise der Künstlichen Intelligenz vertraut gemacht. Nach dem Durcharbeiten des Buches sollte der Leser in der Lage sein, selbständig eigene Problemlösungen in Lisp zu programmieren.

PC-Praxis
Technik und Wissenschaft, Betriebliche Praxis, Benutzerschnittstellen, Betriebssysteme, LAN
von H. Schumny (Hrsg.)
1986, 303 S., 137 Abb, kart., DM 48,–
Vieweg Verlag, Braunschweig
Gliederung
Teil 1, PC in Technik und Wissenschaft: Artikel zu Anwendungen und Erfahrungen mit dem PC in den Bereichen Expertensysteme, Texteditoren und Matrixdrucker,

CAD-Software, Kulisch-Arithmetik, Baumstrukturen, Numerisches Glätten, Interpolieren und Differenzieren, Funktionswertdarstellung, Regelkreisoptimierung, SHARP-Plotter, Beiträge zum Thema „Computer in der Ausbildung" – Teil 2, Betriebswirtschaftliche Praxis: Aufsätze zum Bereich Geschäftsgrafik, Geldanlage, Management mit Beispielanwendungen, Software-Report über kommerzielle Softwarepakete – Teil 3, Benutzerschnittstelle, Programmierung, LAN: Beiträge zur Auswahl und Benutzung verschiedener Systeme, zur Programmierung und Anwendung lokaler Netze (LAN) – Anhang, Produktübersichten: Handheld-Computer (Mobile Computer), Mikrocomputer-Datentabelle, Drucker für Mikrocomputer, Drucker-Datentabellen – Register

Bemerkungen

Dieses Buch, an dem zahlreiche Autoren aus den Bereichen Datenverarbeitung, Elektrotechnik, Wirtschaftswissenschaften und aus dem Ausbildungsbereich mitgearbeitet haben, ist mit den herkömmlichen Sachbüchern nicht direkt vergleichbar. Es ist vielmehr eine Art Lesebuch, das sich in vielen Einzelbeiträgen mit Themen aus der PC-Praxis befaßt. Schwerpunkte bilden dabei die Gebiete

– Technik und Wissenschaft
– Betriebswirtschaft
– Benutzerschnittstellen, Programmierung, Lokale Netzwerke

Abbildungen, Begriffsdefinitionen, Programmbeispiele und Literaturverweise ergänzen die Textbeiträge. Der Anhang enthält eine Produktübersicht über eine breite Auswahl marktgängiger Mikrocomputer und Drucker. In Tabellenform erhält der Leser Auskunft über Hersteller, Bezeichnung, Prozessor, Speicherkapazität, Betriebssystem, Schnittstellen... und Preise. (Letztere sind inzwischen jedoch bereits überholt.) Wer keine speziellen Problemlösungen sucht, sondern sich ein Bild über die Einsatzmöglichkeiten des PCs verschaffen möchte, findet in diesem Buch sicher interessanten Lesestoff.

Praxis der Datenfernübertragung

Ein Wegweiser durch die Datennetze, Mailboxen und Bestimmungen
von A. Pütz
1986, 232 S., 79 Abb., kart., DM 48,–
Franzis Verlag, München

Gliederung

Einführung – Grundlagen – Kommunikationsbranche – Hardware – Software – Datenbanken, Mailboxen – Betriebsablauf – Schaltungen – Tabellen – Anschriften – Begriffe – Literatur – Register

Bemerkungen

Dieses Buch führt den Leser in die Kommunikationswelt der Datenbanken und Mailboxen ein. Neben allgemeinen Informationen über die Grundlagen der Datenfernübertragung und über Kommunikationsnetze enthält das Buch detaillierte Hinweise auf die notwendige Soft- und Hardware. Auf der Hardwareseite werden die auf dem Markt befindlichen Modems und einige wichtige Bauteile besprochen. Schaltungen, die zur Anpassung zwischen dem User-Port und der V.24-Schnittstelle erforderlich sind, und Applikationen der Bauelemente-Hersteller werden im Kapitel 8 ausführlich beschrieben. Mailbox-Verzeichnisse sind im ständigen Fluß, deshalb kann es sein, daß einige der angegebenen Adressen heute bereits nicht mehr aktuell sind. Das Buch ist eigentlich für den Commodore 128 (im 64-Modus) und den C64 geschrieben, doch ist vieles davon allgemeingültig und daher übertragbar. Ein umfangreiches Glossar, Tabellen zum ASCII-Code, den DATEX-P-Knoten und -Netzkennzahlen und den Leitungen der V.24-Schnittstelle runden die Informationen ab.

Reihe Computer verstehen

Grundlagen der Computertechnik

von der Time-Life-Redaktion
1986, 128 S., zahlreiche farbige Abb. und Fotos, geb., DM 44,–
Time-Life Books B.V., Amsterdam

Gliederung

Der Weg zur Computer-Revolution – Die Macht der Binär-Codes – Kriegsbedingte Fortschritte – Die Evolution des Mikrochips – Ein Goldenes Zeitalter des Unternehmertums – Glossar – Bibliographie – Register

Bemerkungen

In der gewohnt aufwendigen Ausstattung der Time-Life-Bücher erschien mit den „Grundlagen der Computertechnik" der erste Band einer geplanten Reihe zum Thema „Computer verstehen". Der Time-Life-Redaktion ist es gelungen, dem Computer-Neuling dieses Thema sehr anschaulich und leichtverständlich nahe zu bringen. Zahlreiche Farbfotos und Ablauf-Darstellungen ergänzen die Textbeiträge. Dieses Buch ist kein Programmierlehrbuch, sondern eine allgemeine Einführung in die Computerwelt. Es stellt die Einsatzbereiche des Mikrochips im Alltag vor, erläutert die Funktionsweise des Computers und liefert einen Abriß über die stürmische Entwicklung dieser Technik von ihren Anfängen bis zur Gegenwart.

Softwareentwicklung auf dem Atari ST

Programmieren unter GEM und TOS
Von Jürgen und Dieter Geiß
1986, 390 S., kart., DM 54,–
Dr. Alfred Hüthig Verlag, Heidelberg

Gliederung

Einleitung – Arbeitsablauf: Editieren, Compilieren, Assemblieren und Linken – Entwicklung von TOS-Programmen – GEM Intern – Entwicklung von GEM-Programmen – Zwei komplette Beispielprogramme – Anhangskapitel – Index

Bemerkungen

Dieses Buch enthält alles, was ein ernsthafter Programmierer braucht, um gute und professionelle Software zu entwickeln. Nach einer Übersicht über die Programmiermöglickeiten des Atari ST erhält der Leser im zweiten Kapitel eine vollständige Erklärung des Arbeitsablaufs sowohl in einer C- wie auch in einer Pascal-Umgebung. Editieren, Compilieren, Assemblieren und Linken werden erklärt, die notwendigen BATCH-Programme sind aufgelistet.

Das Kapitel 3 beschäftigt sich mit der Entwicklung von reinen TOS-Programmen, d.h. Programmen, die als Benutzerschnittstelle die Textoberfläche des Atari verwenden. In diesem Kapitel werden sowohl das Betriebssystem und der Aufruf von BIOS-, XBIOS-, GEM-DOS-Funktionen als auch die Bedeutung der Systemvariablen erklärt.

Kapitel 4 ist das Herzstück dieses Buches: die GEM-Programmierung. Alle Funktionen der beiden großen GEM-Bibliotheken (VDI – Virtual-Device-Interface und AES – Application-Environment-Manager) werden behandelt. Dieser Teil dient später vor allem als Nachschlagewerk. Zwei komplette Sitzungen mit dem Resource-Construction-Set (RCS) werden in Kapitel 5 dargestellt, die daraus resultierenden Objektbäume eingehend demonstriert. Daneben werden wichtige Ratschläge und Strategien zur eigentlichen GEM-Programmierung gegeben. Im letzten Hauptkapitel werden dem Leser in zwei kompletten, sehr sauber programmierten und vollständig kommentierten Beispielprogrammen mit einigen hundert Zeilen fast alle Probleme vor Augen geführt und gelöst, die bei der Fensterprogrammierung (Fenstererweiterung, Dialogboxen, Menüs) entstehen. Diese Programme können als Muster für eigene Applikationen oder Desk-Acessories benutzt werden. Für das schnelle Auffinden von Systemprozeduren und -variablen sind im Anhang ausführliche Tabellen aufgeführt.

MS-DOS

Das optimale Benutzerhandbuch von Microsoft für das Standardbetriebssystem des IBM PC und mehr als 50 andere Personalcomputer
von V. Wolverton
1985, 367 S., kart., DM 78,–
Vieweg Verlag, Braunschweig

Gliederung

Teil 1: Kennenlernen von DOS: Was ist DOS? – Starten von DOS – Probelauf – Ein Blick auf Dateien und Disketten – Teil 2: Umgang mit DOS: Wie Sie Ihre Dateien verwalten – Wie Sie Ihre Disketten verwalten – Wie Sie Ihre Ein-/Ausgabegeräte verwalten – Ein Dateibaum – Wie Sie Ihre Festplatte verwalten – Wie man Textdateien erstellt und editiert – Dateirevision mit Edlin – Wie man selbst Befehle definiert – Wie Sie Ihr System kontrollieren – Wie man elegante Befehle definiert – Weitere elegante Befehlsdefinitionen – Wie Sie Ihr System zurechtschneidern – Teil 3: Kurzbeschreibung der DOS-Befehle: DOS-Befehlsübersicht – Anhangskapitel – Glossar – Sachwortverzeichnis

Bemerkungen

Dieses Buch ist die deutsche Übersetzung der englischen Original-Ausgabe und trägt den Untertitel „Das optimale Benutzerhandbuch von Microsoft für das Standardbetriebssystem des IBM PC und mehr als 50 andere Personalcomputer". Das Buch setzt keine Programmierkenntnisse voraus, erläutert aber nicht, wie DOS arbeitet. Diese Fragen und die Erklä-

rung von Feinheiten der Systemteile (z.B. Drucker oder Monitor) oder seltenere Befehle überläßt es dem DOS-Handbuch. Es enthält dagegen viele anwendungsbezogenen Beispiele, die auf allen MS-DOS-Rechnern laufen. Der erste Teil des Buches beschreibt in 4 Kapiteln die einzelnen Teile des Computersystems, erklärt wichtige Begriffe und zeigt an einigen Beispielen die Vorteile des Betriebssystems MS-DOS. Im Teil 2, dem Hauptteil des Buches, wird der Umgang und die Verwaltung des Computersystems mit Hilfe der DOS-Befehle erläutert: Handhabung der Disketten, Dateien, Computermeldungen, Umgang mit dem Texteditor Edlin, Batch-Befehle, Filterbefehle und Befehlsketten usw. Der dritte Teil ist als Nachschlagewerk konzipiert; jeder DOS-Befehl wird kurz beschrieben, Seitenverweise führen zu Besprechungen und Beispielen in vorausgehenden Kapiteln. Die Anhangskapitel befassen sich mit dem Einsatz einer Festplatte, besonderen DOS-Befehle, dem Unterschied zwischen MS-DOS und PC-DOS von IBM. Ein Glossar und ein ausführliches Register vervollständigen das Handbuch, das im bewährten, leichtverständlichen Stil amerikanischer Lehrbücher geschrieben ist.

CAD

Lehrbuch für die Ausbildung an Fachhochschulen
von G. Hettich
1986, 148 S., kart., DM 44,–
Dr. Alfred Hüthig Verlag, Heidelberg
Gliederung
Einführung – Aufbau von CAD-Systemen: Modelle, Modelltypen, Dimensionalität; Verfahrensfunktionen, Programmteile; Integration bei CAD/CAM-Systemen; Hardware – Bedienung von CAD-Systemen – Einsatz von CAD-Systemen: Entwicklungsvorgang, Konstruktionsarten, CAD-Werkzeuge, Wirtschaftlichkeit – CAD-Praktikum: Elemente und Kommandos eines CAD-Systems, zwei- und dreidimensionale Systeme, Konstruktion – CAD-Studien- und Diplomarbeiten – Definitionen – Literatur – Stichwortverzeichnis
Bemerkungen
Dieses Buch ist auf die Zielgruppe Fachhochschulstudenten ausgerichtet und befaßt sich dementsprechend schwerpunktmäßig mit Fragen der Anwendung und Bedienung von CAD-Systemen. Die Entwicklung dieser Systeme ist Aufgabe von Informatikern und wird in diesem Buch nicht behandelt. Die Vielzahl der verschiedenen CAD-Systeme auf dem Markt macht es notwendig, typenunabhängige Informationen zu liefern. Aufbau und Bedienung von CAD-Systemen werden allgemein beschrieben. Im Kapitel „Einsatz von CAD" werden der Entwicklungsvorgang, die Konstruktionsarten und die adäquaten CAD-Werkzeuge sowie die Probleme der Wirtschaftlichkeit von CAD-Verfahren behandelt. Das Hauptkapitel des Lehrbuches befaßt sich mit der CAD-Anwendung im CAD-Praktikum. Zunächst werden die einzelnen Elemente des CAD-Systems besprochen, dann folgt eine ausführliche, aber allgemein gehaltene und somit für alle CAD-Systeme zutreffende Erläuterung von über 75 verschiedenen Einzelaufgaben aus dem CAD-Bereich. Für Studien- und Diplomarbeiten aus diesem Fachgebiet gibt der Autor einige Anregungen. Ein Glossar und ein ausführliches Literaturverzeichnis dienen zum Nachschlagen und als Anregung für ein vertieftes Studium dieses Faches.

PROLOG

von F. Giannesini, H.Kanoui, R. Pasero und M. van Caneghem
1986, 349 S., kart., DM 58,–
Addison Wesley Verlag, Bonn
Gliederung
Einführung: Theoretische Grundlagen und Programmstruktur – Listen: Listen und endliche Folgen, Teil-Listen, Analyse-Programme – Terme in Präfixform, n-Tupel – Die vordefinierten Regeln von Prolog II: Syntax, Zielkontrolle, Ein- und Ausgabe, Daten, Strukturieren und Modifizieren von Regeln, Programmdurchsicht – Anwendungen: Logik und Datenbanken, Sprachanalyse und -synthese, Unendliche Bäume und Compilation, Automatisches Beweisen und Expertensysteme – Lösungen zu Übungsaufgaben – Anhangskapitel: Sprachsyntax, Prolog-II-Regeln, Entsprechungen Prolog II/ Micro-Prolog bzw. DEC-Prolog – Literatur – Index
Bemerkungen
Prolog wurde schon 1972 von einer französischen Forschergruppe entwickelt und hat ab 1981 in Zusammenhang mit Projekten zur Entwicklung Künstlicher Intelligenz einen starken Aufschwung erfahren. Autoren dieses – in deutscher Übersetzung erschienenen – Buches sind die Entwickler dieser mächtigen Programmiersprache, die dem Leser mit diesem Werk zugleich ein Programmierhandbuch und einen Leitfaden zur „guten" Prolog-Programmierung zur Verfügung stellen möchten. Dabei sieht ihr Konzept eine schrittweise Entwicklung von Beispielprogrammen von der naheliegendsten zur „optimalen" Lösung vor. Um den Prolog-Anfänger nicht zu überlasten, wird die Behandlung der theoretischen Modelle der Sprache weiterführender Literatur überlassen. Die ersten Kapitel über die Grundlagen der Prolog-Programmierung sind nicht an eine bestimmte Prolog-Implementierung gebunden. Die folgenden Abhandlungen benutzen die Version Prolog II, die ein strenges theoretisches Modell, eine klare Syntax und besondere Systemprädikate besitzt. Nach der Einführung in grundlegende Ideen und Mechanismen werden die von Prolog verarbeiteten komplexen Strukturen, die „Listen" und „Bäume" behandelt. Die Grundbegriffe der Verarbeitung formaler Objekte und der syntaktischen Analyse werden anhand von Beispielen erläutert und dann ausführlicher entwickelt. Die vordefinierten Regeln aus Kapitel 5 beschreiben die Schnittstelle von Prolog II zur Umgebung. Das umfangreichste Kapitel befaßt sich mit den Anwendungen von Prolog, d.h. Lösungen zu Problemen der Künstlichen Intelligenz. Dazu gehören strukturierte Datenbanken, Expertensysteme, die Verarbeitung natürlicher Sprachen (wie der Sprache Deutsch) oder klassische Anwendungen wie Compiler. Dieser Teil des Buches ist nur für Fortgeschrittene mit guten Prolog-Kenntnissen gedacht, denn die Methoden stammen z.T. aus aktuellen Forschungsgebieten. Das Buch wird ergänzt durch kommentierte Lösungen zu Übungsaufgaben, einen mehrteiligen Anhang, ein Register und eine Bibliographie.

Schnittstellen-Handbuch

Verständliche Erläuterung und Benutzung von Centronics, V24, IEC-Bus
von J. Elsing und A. Wiencek
1986, 325 S., geb., DM 58,–
IWT Verlag, Vaterstetten
Gliederung
Einleitung – Grundlagen der Datenübertragung – Physikalischer Aufbau von Schnittstellen – Die Centronics-Schnittstelle – Die V.24-Schnittstelle – Der IEC-Bus – Überprüfung von V.24-Anschlüssen – Datenübertragungsbeispiele – Schnittstellen und Datenfernübertragung – Schnittstellenbausteine – Anhang – Literatur – Stichwortverzeichnis
Bemerkungen
Die beiden Autoren möchten mit ihrem Schnittstellenhandbuch die Grundlagen zum Verständnis von Aufbau, Funktion und der praktischen Handhabung der drei Schnittstellentypen Centronics, V24 und IEC-Bus schaffen. Anwendungsbeispiele aus der Praxis – z.B. Bau und Funktion eines einfachen Schnittstellentesters – ergänzen die theoretischen Beschreibungen. Die serielle Schnittstelle und ihre Handhabung (mit Datenübertragungsprogrammen in BASIC, Pascal und Maschinensprache) wird ausführlich behandelt. Die digitalen X.-Schnittstellen und ihre Eigenschaften werden an Beispielen aus der Datenfernübertragung dargestellt. Die wesentlichen Eigenschaften von Schnittstellen wie Steckertypen, Aufbau, Belegungen, Schnittstellenbausteine etc. werden ausführlich erklärt. Enttäuschend ist die typographische und gestalterische Ausstattung des Buches. Es ist in einfacher Schreibmaschinenschrift gehalten, wichtige Begriffe, neue Abschnitte und die Hauptkapitel des Buches werden nicht gesondert hervorgehoben. Für DM 58,– dürfte man als Leser eine bessere Ausstattung erwarten.

Leserbriefe

CP/M 3.0 beim Apple II

Zuerst möchte ich zu dem hervorragenden Artikel „CP/M 3.0 beim Apple II" im Peeker 11/86, S. 34ff. gratulieren. Wer schon mit CP/M 3 gearbeitet hat, wird CP/M 2.2 kaum vermissen, außer wenn er die Grafikmöglichkeiten via 6502 nutzen möchte, was bisher offenbar unter CP/M 3 nicht möglich ist.

Einiges zu diesem Artikel bedarf jedoch noch einer Ergänzung. So erhielt ich zu meinem CP/M-3-Paket für den Basis 108 nicht nur die von den Autoren erwähnten User's Guide, Programmer's Guide und System Guide, sondern auch noch der Programmer's Utilities Guide (256 Seiten) sowie der SID User's Guide (72 Seiten). Zudem sind die CP/M-3-Möglichkeiten mit dem Basis 108 noch um einiges größer als mit dem Apple. Man denke da nur an die 15 Funktionstasten, die mit CP/M-3-Befehlen belegt werden können u.a.m. Die erwähnten Editierfunktionen sind natürlich zusätzlich auf den separaten Cursortasten mit erweiterten Möglichkeiten untergebracht.

Als Nachteil wurde der Umgang mit dem offenbar vertrauten SAVE-Befehl genannt. Das beschriebene, komplizierte Vorgehen mit dem SAVE und SID ist jedoch nicht notwendig. Noch mehr, SAVE ist mit dem Programm SID überflüssig geworden und wird mit dem CP/M-3-Paket oft gar nicht mehr geliefert, obwohl es im User's Guide noch beschrieben ist. Unter den ca. 20 SID-Kommandos existiert ein Schreibbefehl, der den Speicherinhalt auf die Diskette überträgt. Das umständliche Ermitteln der Speicherlänge, wie es bei SAVE nötig war, entfällt. SID ermittelt sie automatisch. Als Option kann noch die Start- und/oder End-Adresse des zu speichernden Inhalts angegeben werden.

Getestet wurde von den Autoren offenbar auch Turbo-Pascal unter CP/M 3. Hier sind einige Einschränkungen anzubringen. So zeigen bisher alle Turbo-Pascal-Versionen 2.00A/3.00A/3.01A unter dem DIR-Befehl eine falsche Größe des noch vorhandenen freien Speicherplatzes auf der Diskette an. Dies wird vor allem ersichtlich, wenn Diskettenlaufwerke größerer Kapazität (z.B. Erphi) verwendet werden. Mit einem Patch kann jedoch Turbo-Pascal auf das CP/M 3 angepaßt werden.

Zudem funktionieren die bios()- bzw. bioshl()-Anweisungen unter CP/M 3 nicht mehr und müssen über eine der erweiterten BDOS-Funktionen (BDOS 50) umgeleitet werden. Ob dieses Fehlverhalten in der neuesten Turbo-Pascal-Version 3.01A immer noch besteht, habe ich nicht erprobt. Daher müssen Turbo-Pascal-Programmierer darauf achten, wenn sie ein einwandfreies Arbeiten unter CP/M 2 und CP/M 3 wünschen. Am einfachsten zu realisieren ist es mit der Abfrage der CP/M-Version (BDOS 12) und den somit entsprechenden Befehlen.

Rolf Gachnang, Rapperswil (Schweiz)

Kyan Pascal

Im Kyan Pascal System habe ich zwei Mängel entdeckt, über die ich kurz berichten möchte.

1. Der Compiler macht sich scheinbar keinerlei Gedanken über den für Variablen zur Verfügung stehenden Speicherplatz. Dies zeigt das folgende kurze Programm:

```
PROGRAM TEST(INPUT,OUTPUT);
CONST DIM= 17000;
VAR
I:INTEGER;
AR:ARRAY [1..DIM] OF INTEGER;

BEGIN
WRITELN('WRITING..');
FOR I:= 1 TO DIM DO AR[I]:=I;
WRITELN('READING..');
FOR I:= 1 TO DIM DO
BEGIN
WRITE(I,' ');
IF AR [I] <> I
THEN WRITELN('ERROR!');
END
END.
```

Bei DIM = 16000 gibt es keine Probleme, bei DIM = 17000 stürzt das Programm in den Monitor ab und bei DIM = 18000 gibt es eine Runtime-Fehlermeldung und Unfug auf dem Bildschirm. Bei einem anderen Programm geschah es einmal (in Verbindung mit '_USES-HIRES'), daß der Inhalt von Variablen während des Programmablaufs unkontrolliert verändert wurde.

2. Die Prozedur readln akzeptiert bei der Eingabe von Fließkommazahlen nur ein kleines „e" für den Exponenten. Bei Eingabe von der Tastatur ist das zwar lästig, aber nicht weiter schlimm (wenn man es weiß). Problematisch ist der Fehler nur im Hinblick auf die Benutzung von readln beim Einlesen von einer Textdatei, die man zuvor mit writeln geschrieben hat: writeln schreibt die Fließkommazahl mit einem großen „E" auf die Datei, das führt beim Einlesen mit readln aber dazu, daß immer „-9.0E-99" eingelesen wird. Dies ist lästig, wenn man für immer wiederkehrende Eingaben für ein Programm eine Textdatei anstelle der Tastatureingabe benutzt, und die Eingaben nicht nur aus Real-Zahlen bestehen, so daß „File of real" nicht anwendbar ist.

Dr. S. Stieler, Gießen

Apple II an Gymnasien

97% aller Gymnasien in Rheinland-Pfalz betreiben „informationstechnische Grundbildung" und „Informatik in der Sekundarstufe II" mit Apple-II-Computern. Dies liegt zum großen Teil daran, daß dies der erste verfügbare Rechner zwischen 1976 und 1978 war, mit dem man Pascal programmieren konnte (Pascal wird ab 1976 in der „Mainzer Studienstufe" von Rheinland-Pfalz gelehrt). In einer Zeit, in der alle zwei Jahre die gesamte technische Entwicklung überholt ist, betreiben wir noch zum größten Teil diese vor zehn Jahren angeschafften Rechner im härtesten Einsatz, der für Unterrichtsmaterial denkbar ist (Unterricht, offener Laborbetrieb, Lehrerausbildung). Die Einführung des Apple IIe lag zu einem Zeitpunkt, zu dem die Computerausrüstung der Schulen erst größere Dimensionen annahm, so daß dies der am weitesten verbreitete Rechnertyp (wohl auch in Nordrhein-Westfalen) ist. Diese ohne staatliche Lenkung doch recht einheitliche Entscheidung ist wohl noch immer zu unterstützen, auch wenn es inzwischen Rechnersysteme am Markt gibt, deren Leistungsfähigkeit bei gleichem Preis erheblich größer ist. Dies liegt daran, daß man heute einen didaktisch sinnvollen, allgemeinbildenden und zukunftsorientierten Informatik-Unterricht (ich beziehe mich auf das Gymnasium) in völlig ausreichender Form mit dieser Rechnerleistung durchführen kann. Es benötigt heute in der Schule niemand einen schnelleren, speicherplatzvergrößerten, grafikverbesserten, geschweige denn einen besser softwareversorgten, vernetzten oder gar teureren Rechner (die Verwaltung einmal ausgenommen)! Ja, man könnte sogar soweit gehen, von Steuermittelverschwendung bei größeren Systemen zu sprechen! Solange unsere Rechner halten (was unter den Bedingungen erstaunlich lange zu sein scheint), werden wir wohl weitere Generationen an diesen Rechnern ausbilden!

A. v. Irmer, Bolanden

Umwandlung von Großbuchstaben bei dBase II

Antwort auf die Frage von Peeker-Leser Ch. Waller aus Frankfurt (Peeker 1/87, S.70): Die Umwandlung in Großbuchstaben bei dBase II läßt sich durch die folgenden Änderungen erreichen, außerdem wird die Eingabe von „ß" zugelassen:

Im File DBASE.COM müssen geändert werden

12F5H von 7B in 7E
3688H von 5B in 5E
368FH von 7B in 7E

Am besten mit DDT ändern und anschließend mit SAVE 76 DBASENEU.COM auf Diskette abspeichern. Aus: Axel Unterschütz, dBase 2.4 mit deutschen Sonderzeichen, in: c't Special 1, Software Know-How.

J. Neßelrath, Düren

Ist Sprache eine Metapher?

Ich habe mit Vergnügen diesen Beitrag von U. Stiehl (Peeker 1/87, S.42) gelesen. Beiträge dieser Art sind es, die Fachzeitschriften weniger langweilig machen und möglicherweise Software-Autoren zu unkonventionellen Ideen veranlassen. Sicher war es mühsam, in der Weltliteratur diese Sammlung von Sprachdefinitionen zu finden, die allesamt hart an der Schmerzgrenze liegen. Doch die Mühe lohnte sich. Nach den vorhergehenden Definitionen ist der Leser leicht geneigt, die angebotene Definition völlig kritiklos hinzunehmen.

Bei der brillanten Reduktion dieser Definition auf das Wesentliche hat Herr Stiehl am Schluß jedoch Mühe, daß überhaupt etwas übrig bleibt. Der Grund könnte sein, daß Sprache primär kein Mittel zur Kommunikation (im Sinne von Gemeinsam-Machung) ist.

Versuchen wir es doch einmal mit mathematischen Mitteln: Sprache ist die umkehrbar eindeutige Transformation einer Menge abstrakter Inhalte (nicht nur des Bewußtseins) auf eine endliche Menge von Symbolen (Wortschatz) zwecks Vereinfachung der Übertragung. Eine definierte Syntax ist nicht notwendiger Bestandteil einer Sprache; sie erlaubt lediglich eine kürzere Darstellung oder eine Verringerung des Wortschatzes. So ließe sich im vorgegebenen Beispiel „Hans sieht Fritz.." der sehende Fritz vom gesehenen Fritz auch durch unterschiedliche Symbole anstelle der syntaktischen Reihenfolge unterscheiden. Diese Definition unterscheidet sich von der von Herrn Stiehl hauptsächlich durch das Ziel der Vereinfachung. Dennoch deckt sie Sprache vom Kasernenhofgebrüll bis zur Lyrik und sogar Programmiersprachen voll ab: Ohne Sprache müßte der Ausbilder seinen Rekruten vormachen, wie man sich in ein Schlammloch wirft; und damit ist er weit besser dran als der Lyriker, der ohne Sprache wohl ganz auf die Vermittlung seiner Gefühle verzichten müßte. Die Programmiersprache hat mit beiden Extremen Gemeinsamkeiten: Stil und Syntax sind dem Kasernenhofgebrüll sehr ähnlich, was nicht verwundert, da es sich ebenfalls um eine Vereinfachung der Befehlsübertragung handelt. Ähnlich wie bei der Lyrik geht jedoch ohne Sprache rein gar nichts. Auch das verwundert nicht, denn die Kommunikation zwischen zwei Wesen unterschiedlicher Gefühlsstruktur ist ebenso problematisch wie die Übertragung der Vielfalt menschlicher Wünsche auf ein so beispiellos stumpfsinniges Gebilde wie es ein Rechner nun mal ist.

Tut man sich ohne das Ziel der Vereinfachung schwer, Programmiersprachen als Sprachen zu akzeptieren, so drängen sie sich mit diesem Ziel geradezu als Beispiel auf. Ein Programmierer formuliert die dem Rechner abverlangte Reaktion in einer vergleichsweise einfach zu erlernenden Sprache. Ein je nach Arbeitsweise Interpreter oder Compiler genannter Dolmetscher übersetzt diesen Text in eine weit einfachere, dem Verständnis des Programmierers noch fremdere Maschinensprache. Aber auch diese Sprache dient in erster Linie Menschen, z.B. den Autoren von Interpretern und Compilern. Um eine Ausführung im Prozessor zu erreichen, übersetzt das Mikroprogramm des Prozessors den Maschinencode in eine aus zwei Zeichen ohne Syntax bestehende und nicht weiter zu vereinfachende Sprache.

Genau hier bin ich an einer Stelle angelangt, wo mir deutlich wird, welchen Nutzen es haben kann, sich auch einmal gedanklich an der Rändern der Computerei zu bewegen. Unsere Muttersprache wurde seit Generationen optimiert, und zwar hinsichtlich eines Minimums an Gesamtaufwand zum Erlernen und zum Kommunizieren innerhalb eines Menschenlebens. Eine Sprache mit umfangreichem Wortschatz und Regelwerk erlaubt eine kurze Abfassung von Inhalten, ist aber schwer zu erlernen – und umgekehrt. Über diesen Gegensatz hinaus muß bei einer Programmiersprache noch die Anpassung an den Rechner zwecks Erhöhung der Verarbeitungsgeschwindigkeit berücksichtigt werden.

Erste Ansätze für solche Optimierungen zeigen sich in Begriffen wie „rechnerorientierte" oder „problemorientierte" Programmiersprache. Den Begriff „menschenorientierte" Programmiersprache habe ich leider noch nicht gehört. Ein erster Ansatz für die Berücksichtigung des Menschen ist das „B" für Beginners im Kürzel „BASIC". Allerdings drängt sich bei näherer Betrachtung der Verdacht auf, daß es sich

bei „Beginners" nicht um Anwender, sondern um die Autoren handelt, denn es ist schwer einzusehen, warum BASIC für Anfänger besser geeignet sei als all die anderen für Mensch und Maschine gleichermaßen unverständlichen Sprachen von ALGOL bis C.
Leider werden Rechner von Technikern erdacht – im günstigsten Fall sind höchstens Informatiker daran beteiligt – in jedem Fall also Leute, die einfach nichts Besseres gelernt haben. Dabei gibt es bereits leistungsfähige Prozessoren wie etwa die 68000-Reihe, deren Befehlssatz so simpel ist, daß sich ein mittelmäßig begabter Sprachwissenschaftler auf dieser Spielwiese austoben könnte.
Am Schluß ein Empfehlung für den Peeker: Grasen Sie weiterhin an den Rändern der Computer-Spielwiese; es ist eine der wenigen Möglichkeiten, sich als Fachjournal heute noch zu profilieren.
R. Ranfft, Haan-Gruiten

Der Artikel „Ist Sprache eine Metapher?" von U. Stiehl beschreibt zutreffend die derzeitige Situation. Jedoch rechtfertigt sein grundsätzlicher Anspruch wohl einen ebenso grundsätzlichen Kommentar. Die Bindung von Sprache = Intelligenz an menschliche Gehirne wird als Selbstverständlichkeit vorausgesetzt (im folgenden unterscheide ich nicht zwischen Sprache und Intelligenz; der Platz ist zu knapp). Wir müssen uns aber hüten, immer wieder in eine der Fallen zu gehen, aus deren erster uns Kopernikus befreit hat. Letztere Formulierung ist zum Beispiel immer wieder von Hoimar v. Ditfurth gebraucht worden; im folgenden zitiere ich, was er zur vorliegenden Frage zu sagen hat (1):
„<Wir wissen>, daß die Evolution schon auf molekularer Ebene, noch vor dem Erreichen des biologischen Abschnittes ihrer Geschichte, prüft, auswählt, bewertet, ausprobiert und nach Problemlösungen sucht..." (Zuvor hat er den Mechanismus dieser evolutionären Intelligenz aufs Ausführlichste beschrieben). „<Diese> Beispiele beziehen sich auf...jene Entwicklung..., die schließlich auch unser Gehirn hervorgebracht hat..." „Die unbestreitbare Hirnlosigkeit der Natur <scheint uns aber> gleichbedeutend mit der Nicht-Existenz von Intelligenz, Phantasie, Lernfähigkeit..." „Ich fürchte, daß ein nicht unwesentlicher Teil unseres Staunens über die Natur auf einem Mißverständnis beruht, das hier seine Wurzeln hat...Als ob der ganze Kosmos, jahrmilliardenlang, ohne all die genannten Fähigkeiten hätte auskommen müssen, weil es uns noch nicht gab..." „Dafür, daß Hirnlosigkeit nicht in jedem Falle mit dem Fehlen von Intelligenz gleichzusetzen ist..., genügt schon unsere Leber als Beweis." „<Das alles> darf auf gar keinen Fall als ein Rückschluß auf die Aktivität einer übernatürlichen Weisheit welcher Art auch immer mißverstanden werden."
Es folgen einige Betrachtungen, ob Computer intelligent (sogar „bewußt"?) werden könnten, besonders die spätere Anmerkung 133.
(1) v.DITFURTH, Hoimar (1984): „Wir sind nicht nur von dieser Welt", dtv München, S. 265-269.
Dr. H. Benda, Ismaning

RAM-Disk-Treiber für Saturn-Karte

Ich habe bereits länger nach einem RAM-Disk-Treiber, der unter PRODOS läuft, für die Saturn-128K-RAM-Karte der Firma TITAN gesucht. Durch die Mithilfe der Firma Weiss in Wilhelmshaven habe ich in den USA die gesuchte Software gefunden. Der RAM-Disk-Treiber ist sehr einfach zu installieren und hat meine Erfahrungen mehr als erfüllt! Das Programmieren mit Kyan-Pascal macht jetzt riesig Spaß. Der Preis für diese Software inkl. Versand und Verpackung kostete mich ca. US$ 20,–. Diese Nachricht würde sicher manchen Apple-Besitzer interessieren, der mit Kyan-Pascal arbeitet und eine Saturn-Karte besitzt. Deshalb hier die Adresse des Autors:
Steven Humpage
2427 NE 24th Ave
Portland, OR 97212
USA
Jürg Flacher, Winterthur (Schweiz)

Errata

Patch für ProDOS-Fast-Writer

Wegen eines Problems mit ProDOS gibt es Schwierigkeiten beim Laden und Speichern der Tab-Leiste-Datei. Mit dem folgenden Patch-Programm läßt sich das Problem beheben:

```
10 PRINT CHR$(4)"BLOAD FW.SYSTEM,TSYS,A8192"
20 POKE 8645,254: REM BIT-MAP
30 PRINT CHR$(4)"BSAVE FW.SYSTEM,TSYS,A8192,L13320"
```

Produkte

CAPascal – ein Software-Werkzeug für die Schule

Messen, Steuern und Regeln ist zu einer der Lerninhalte in der Schule sowohl in der Informatik als auch in anderen Fächern (wie Physik) geworden. Auf dem Markt befindliche Interface-Systeme für Apple-II-Computer werden schon vielfältig angeboten, so z. B. das „CAP-Interface" der Lehrmittelfirma Leybold. Leider werden zu diesem System lediglich für bestimmte Problemlösungen fertige BASIC-Programme geliefert (was für das Gymnasium jedoch weitgehend uninteressant ist). Diese Programme sind für einen mittelguten Programmierer nur schlecht zu ändern oder gar in eigene Problemlösungen zu integrieren, zudem diese Programmiersprache in der Oberstufe wohl landesweit keine Rolle spielt.
CAPascal ist ein in die Bibliothek des UCSD-Pascal-Systems eingebettetes Software-Werkzeug, mit dem die Möglichkeit besteht, sämtliche Funktionen des CAP-Interface (auf der Sprachebene Pascal) zu programmieren (z.B. INITCAP, RELAIS(ON), AD_CONVERSION usw.). Da die Routinen in Assembler geschrieben sind, gelingt es mit CAPascal, wesentlich zeitoptimierter zu arbeiten. Für stark zeitkritische Arbeiten (z.B. Funktionen der eingebauten Timer) gibt es noch die Möglichkeit, in eigene Assemblerprogramme fertige Makros einzubauen. Der auf der Rückseite des Gerätes herausgeführte Port-A des VIA-6522-Bausteins-I ist ebenfalls softwaremäßig implementiert.
Informationen: Arved von Irmer, Am Schützenpfad 15, 6719 Bolanden, Tel. 06352/8765.

Programming Toolkits

für Kyan-Pascal 2.0

Toolkit I: System Utilities: Club-Preis DM 118,-; Normalpreis DM 148,-
Toolkit II: Mouse Text: Club-Preis DM 118,-, Normalpreis DM 148,-
Toolkit III: Advanced Graphics: Club-Preis DM 118,-, Normalpreis DM 148,-
Toolkit IV: Turtle Graphics: Club-Preis DM 68,-, Normalpreis DM 88,-
Toolkit V: Mouse Graphics: Club-Preis DM 158,-, Normalpreis DM 198,-

Alle Utilities werden als teils beidseitig bespielte Disketten geliefert, die neben den Include-Files (meist Quelltexte) diverse Demos enthalten. Die Anleitungen selbst sind Loseblattlieferungen (z.B. bei den System Utilities 52 Druckseiten), die für den grauen Ordner von Kyan 2.0 bestimmt sind. Zum Club-Preis werden nur Mitglieder des Kyan-Clubs beliefert.

Toolkit I: System Utilities

Diese Utilities decken verschiedene Bereiche ab:
1. ProDOS-Utilities: Delete, Rename, Copy, Set-Pefix, Get-Prefix, Lock, Unlock, Make-Directory, Get-Directory, Remove-Directory, Scan-File, Format, Print-File, Bsave, Bload, Set-Time, Get-Time, Set-Date, Get-Date, Get-Clock, Set-Clock, Find-Clock.
2. Maus und Joystick: Find-Mouse, Init-Mouse, End-Mouse, Home-Mouse, Mouse-Click, Mouse-held, Mouse-moved, Mouse-x, Mouse-y, Set-Mouse-xy, Set-x-Bounds, Set-y-Bounds, Print-Mouse-Char, Button-0, Button-1, Joystick-x, Joystick-y.
3. Bildschirmsteuerung (funktionieren teilweise nur auf IIe/c): Clear-Screen, Clear-Line, Clear-End-of-Line, Clear-End-of-Page, Inverse, Normal, Tab, Scroll-up, Scroll-down, Col-80, On-40, On-80, Screen-Bottom, Screen-Top, Screen-Full, Cursor-x, Cursor-y, Get-Char, Machine-Identification.
4. Zufallszahlen: Random im Bereich min..max, Rnd im Bereich 0..1, Seeding.
5. Zahlenkonvertierungsroutinen: Real – String, String – Real, Integer – String, String – Integer.
6. Sortieren (alphabetisch und numerisch) sowie Mischen (bis zu 5 Files).
7. Line Parsing Routine.

Toolkit II: Mouse Text

Diese Utilities umfassen mehrere Dutzend Befehle für Fenstertechnik usw.
1. Cursor-Befehle: Set-Cursor, Obscure-Curser, Hide-Cursor, Show-Cursor.
2. Interrupts: Check-Events, Get-Event, Post-Event, Set-Key-Event, Flush-Event, Peek-Event.
3. Menü-Befehle: Init-Menu, Set-Menu, Menu-Select, Menu-Key, High-Light-Menu, Disable-Menu, Disable-Item, Check-Item, Set-Mark.
4. Kontrollbefehle: Find-Control, Set-Control-Max, Track-Thumb, Update-Thumb, Activate-Control.
5. Fensterbefehle: Init-Window-Margin, Close-Window, Open-Window, Find-Window, Front-Window, Select-Window, Drag-Window, Grow-Window, Screen-to-Window, Window-to-Screen, Close-all, Window-Char, Window-String, Window-Block, Window-Text, Window-Op.

Toolkit IV: Turtle Graphics

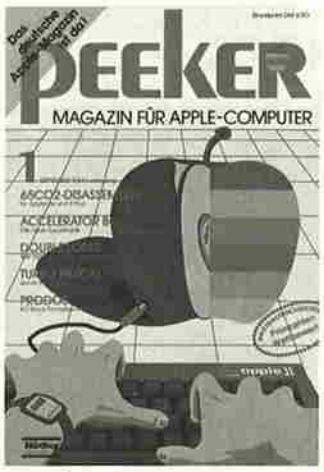
Diese Diskette enthält diverse Hires- und Ton-Routinen.
1. Turtle-Befehle: Init-Turtle, Turtle-x, Turtle-y, Turtle-Angle, Graf-Mode, Text-Mode, Pen-Color, Turn, Turn-to, Move, Move-to, View-Port, Full-Port, Fill-Port, Fill-Area, Save-Hires, Load-Hires.
2. Ton-Befehle: Beep, Note, Clock, Phaser.
3. Balkendiagramme usw.: Bar-Chart, Pie-Chart, Plot-x-y.

Nur lieferbar, solange Vorrat reicht!

Hüthig Software Service
Postfach 102869 Heidelberg

Peeker-Sammeldisketten

Gesamtverzeichnis #1 bis #27



Sammeldisk #1

```
A ØØ2 PEEKER ═══ HEFT 1/84 ═══
A ØØ2 A-
T Ø29 T.DISASSEMBLER.65CØ2
B ØØ4 DISASSEMBLER.65CØ2
A ØØ2 B-
T ØØ4 T.ACCEL.WAIT
B ØØ2 ACCEL.WAIT
T ØØ8 T.ACCEL.BOOT
B ØØ2 ACCEL.BOOT
A ØØ2 ACCEL.LC.KOPIERER
T ØØ8 T.ACCEL.LC.KOPIE
B ØØ2 ACCEL.LC.KOPIE
T Ø1Ø T.ACCEL.ROM.KOPIE1
B ØØ2 ACCEL.ROM.KOPIE1
T ØØ8 T.ACCEL.ROM.KOPIE2
B ØØ2 ACCEL.ROM.KOPIE2
A ØØ2 C-
A Ø27 TURTLE GRAFIK MIT REMS
A Ø18 TURTLE GRAFIK OHNE REMS
A ØØ2 D-
A ØØ3 DOUBLE.LORES.SOFTSWITCH.DEMO
A ØØ4 DOUBLE.LORES.APPLESOFT.DEMO
A ØØ3 AMPER.DOUBLE.LORES.DEMO
T ØØ9 T.AMPER.DOUBLE.LORES
B ØØ2 AMPER.DOUBLE.LORES
T Ø11 T.DOUBLE.LORES
B ØØ2 DOUBLE.LORES
A ØØ2 E-
A Ø18 HIRES
T Ø4Ø T.PRINTHIRES
B ØØ5 PRINTHIRES
A ØØ2 PEEKER ═══ HEFT 2/84 ═══
A ØØ2 F-
A ØØ7 DHGR.APSOFT.DEMO
A ØØ3 AMPER.DOUBLE.HIRES.BAS
B ØØ4 AMPER.DOUBLE.HIRES
T Ø15 T.AMPER.DOUBLE.HIRES
A ØØ8 DHGR.LINEPLOTTER
A ØØ2 G-
A ØØ3 INSTRING.TEST
B ØØ2 INSTRING.OBJ
T Ø19 T.INSTRING.OBJ
B Ø14 INSTRING.LISA.SOURCE
A ØØ2 H-
A ØØ4 LOESCHEN.EINES.ARRAYS
A ØØ2 I-
B Ø18 ULTRATERM.ENGLISCH
B Ø18 ULTRATERM.DEUTSCH
A ØØ2 J-
A ØØ4 PRIMZAHLEN.OVERMEYER
B ØØ2 PRIM.OBJØ
B ØØ2 PRIM.OBJ1
A ØØ2 PRIM.TEST
T Ø12 PRIM.TOOLKIT.SOURCE
```

Sammeldisk #2

```
A ØØ2 PEEKER ═══ HEFT 1-2/85 ═══
A ØØ2 A-
T Ø31 T.RAMDISKLC
B ØØ4 RAMDISKLC
T Ø12 T.IBS.RAMDISKDRIVER
B ØØ3 IBS.RAMDISKDRIVER
T ØØ6 T.AP2Ø.RAMDISKTEST
B ØØ2 AP2Ø.RAMDISKTEST
A ØØ2 B-
T Ø26 T.QUICKCOPY
B ØØ9 QUICKCOPY
A ØØ3 QUICKCOPY.PUFFER
A ØØ4 PRODOS.COPYA
T Ø19 T.PRODOS.COPYOBJ
B ØØ7 PRODOS.COPYOBJ
A ØØ9 PRODOS.PATCH
A ØØ2 C-
T Ø23 T.APPLESOFT.FRE
T Ø32 T.LC.FRE
B ØØ3 LC.FRE
A ØØ3 FRE.TEST
T Ø24 T.RAM.FRE
B ØØ3 RAM.FRE
A ØØ2 D-
T Ø1Ø T.SCHIRMDISK
B ØØ7 SCHIRMDISK.LISA.SOURCE
B ØØ2 SCHIRMDISK
T Ø12 T.VIDEXT
B ØØ9 VIDEXT.LISA.SOURCE
B ØØ2 VIDEXT
A ØØ2 E-
A Ø1Ø GETPAS
T Ø11 T.GETPAS.ASS
B ØØ2 GETPAS.ASS
T Ø33 GETDOS.PASCAL.SOURCE
T Ø22 COPYDUPDIR.PASCAL.SOURCE
A ØØ2 F-
A ØØ5 PRODOS.EDITOR.MACROS
```

Sammeldisk #3

```
CP/M - MBASIC
Einkommensteuerprogramm

PASS.BAS
A.BAS bis N.BAS
MENUE.BAS
HELP.BAS
```

Sammeldisk #4

```
A ØØ2 PEEKER ═══ HEFT 3/85 ═══
A ØØ2 A-
A Ø21 TESTGENERATOR
T ØØ3 BAHNFAHRT
T ØØ2 ZU
T ØØ3 TUN.UND.SOLLEN
T ØØ2 IRGEND
T ØØ3 SAETZE
A ØØ2 B-
A Ø17 MULTIPRECISION
A ØØ2 C-
T Ø46 T.WS.TRANSFER
B ØØ8 WS.TRANSFER
T Ø5Ø T.WS.TRANSFER.2
B ØØ9 WS.TRANSFER.2
A Ø1Ø GETCPM
A ØØ2 D-
I Ø11 PRIM.Ø.SC.SOURCE
B ØØ2 PRIM.Ø.BIN
I Ø15 PRIM.1.SC.SOURCE
B ØØ3 PRIM.1.BIN
A ØØ7 PRIM.FP
A ØØ2 E-
A ØØ3 ACCELERATOR.ABSTELLEN
A ØØ2 F-
T ØØ5 T.WILDCARD.TEST1
B ØØ2 WILDCARD.TEST1
T ØØ5 T.WILDCARD.TEST2
B ØØ2 WILDCARD.TEST2
A ØØ2 PEEKER ═══ HEFT 4/85 ═══
A ØØ2 G-
A ØØ3 XPLOT.DEMO
B ØØ2 XPLOT.ROUTINE
T ØØ8 T.XPLOT.ROUTINE
A ØØ2 H-
A Ø22 MENUE.GENERATOR
A ØØ2 I-
T Ø24 T.MACROS.65CØ2
A ØØ2 J-
A Ø22 TERMINAL
B ØØ6 TERMINAL.B
T Ø32 T.TERMINAL.B
A ØØ2 K-
A ØØ3 CAT.ARRAY
A ØØ3 CAT.SAVER
A ØØ3 EINTRAG.SUCHER
A Ø1Ø EINTRAG.ANALYSE
A ØØ6 PRODOS.READER
T ØØ4 T.PRODOS.READER.OBJ
B ØØ2 PRODOS.READER.OBJ
A ØØ2 L-
T Ø12 MOUSESTUFF.PASCAL.SOURCE
T Ø5Ø MOUSE.ASS.PASCAL.SOURCE
T ØØ7 TESTMOUSE.PASCAL.SOURCE
T Ø42 DRAWMOUSE.PASCAL.SOURCE
T ØØ1 M-
A ØØ2 INALL.DATA
A ØØ4 SCREEN8Ø.DATA
A ØØ3 SCREEN8Ø.SAVER
```

Sammeldisk #5

```
A ØØ2 PEEKER ═══ HEFT 5/85 ═══
A ØØ2 A-
T Ø14 T.FM.BSP
B ØØ3 FM.BSP
A ØØ2 B-
T Ø37 T.SLOTRAMDISK
B ØØ5 SLOTRAMDISK
A ØØ2 SLOTRAMDISK.HELLO
```

```
A ØØ2 C-
A Ø54 PLOT.2.Ø
T Ø15 T.PLOT.B
B ØØ3 PLOT.B
T ØØ2 PLOT.PROTECTOR
A ØØ2 D-
T Ø14 T.CONVERT56Ø
B ØØ2 CONVERT56Ø
A ØØ2 CONVERT56Ø.DEMO
A ØØ2 E-
T Ø18 T.EDA
B ØØ3 EDA
A ØØ2 F-
T Ø36 TRANSCEND.PASCAL.SOURCE
A ØØ2 G-
T ØØ6 T.BLOCKTRACER
B ØØ2 BLOCKTRACER
T ØØ6 T.BLOCKTRACER1
B ØØ2 BLOCKTRACER1
A ØØ2 H-
B Ø14 FORMAT.LC
A ØØ3 FORMAT.LC.START
T ØØ7 T.DISKDRIVER.DEMO
B ØØ2 DISKDRIVER.DEMO
A ØØ2 I-
A ØØ8 RANDOM.DEMO
A ØØ4 COLUMN8Ø.DEMO
A Ø24 SUPERDUMP.EPSON
A Ø24 SUPERDUMP.IMAGEWRITER
B Ø66 SUPERDUMP.BILD
T 116 T.SUPERDUMP
B ØØ8 SUPERDUMP
B ØØ3 EPSON
B ØØ3 IMAGEWRITER
```

Sammeldisk #6

```
A ØØ2 PEEKER == HEFT 6/85 ==
A ØØ2 A-
A Ø24 HELLO
B 114 ASMDIV
A ØØ2 B-
B ØØ2 CURSOR1
T ØØ4 T.CURSOR1
B ØØ2 CURSOR2
T ØØ4 T.CURSOR2
B ØØ2 LINIE
T ØØ5 T.LINIE
B ØØ2 VIERECK
T ØØ5 T.VIERECK
B ØØ2 BOX
T ØØ9 T.BOX
B ØØ2 HINTERGRUND
T ØØ5 T.HINTERGRUND
B ØØ2 PAGE.SWAP
T ØØ4 T.PAGE.SWAP
A ØØ2 C-
A ØØ3 WANDERNDER.STRICH
A ØØ4 KOMPRESSOR.DEMO
A ØØ3 KREIS.1
A ØØ4 KREIS.2
A ØØ4 KREIS.3
B ØØ2 FLIPPER
T ØØ5 T.FLIPPER
B ØØ2 KOMPRESSOR
T Ø12 T.KOMPRESSOR
A ØØ2 D-
B ØØ3 OLYMPIA
T Ø21 T.OLYMPIA
A ØØ2 E-
A Ø43 FOURIER.MAIN
A Ø19 FOURIER.SYN
A Ø2Ø FOURIER.SPEC
A ØØ2 F-
B ØØ4 AS.ERWEITERUNG
T Ø28 T.AS.ERWEITERUNG
A ØØ3 AS.ERW.PROSTART
B ØØ4 AS.ERW.PRO
T Ø17 T.AS.ERW.PRO
A ØØ2 G-
T ØØ2 INSTALL.PASCAL.SOURCE
T ØØ5 RAMDISK94.PASCAL.SOURCE
T Ø32 INIT.PASCAL.SOURCE
A ØØ2 H-
A ØØ2 RAMDISK.INIT.DOS
B ØØ2 AUXDRIVER
T Ø1Ø T.AUXDRIVER
B ØØ2 MOVEDRIVER
T ØØ4 T.MOVEDRIVER
B ØØ2 RAMDISK.FORMATTER
T ØØ6 T.RAMDISK.FORMATTER
A ØØ2 J-
A ØØ2 SOLITAIRE.START
A Ø29 SOLITAIRE
B ØØ4 SOLITAIRE.B
T Ø16 T.SOLITAIRE.B
```

Sammeldisk #7

```
A ØØ2 PEEKER == HEFT 7/85 ==
A ØØ2 A-
A ØØ4 PYRAMID.PITTY
T Ø84 T.PYR.PITTY.Ø
T Ø63 T.PYR.PITTY.1
B Ø18 PYR.PITTY.Ø
B Ø16 PYR.PITTY.1
B Ø24 PYR.PITTY.BACK
B Ø26 PYR.PITTY.SHAPE
A ØØ2 B-
T Ø16 T.MEGAWARP.REL
B ØØ2 MEGAWARP.REL
T Ø17 T.MEGAWARP.99ØØ
B ØØ2 MEGAWARP.99ØØ
T ØØ4 T.SPEEDTEST
B ØØ2 SPEEDTEST
A ØØ2 C-
A Ø16 FORMAT
T Ø39 T.FORMAT.OBJ
B ØØ5 FORMAT.OBJ
A ØØ2 D-
A Ø33 BITEDITOR
B Ø18 NORMAL
B Ø18 FETT
B Ø18 FETT.INVERSE
A ØØ2 E-
A ØØ7 PASTOPRO.1D
A ØØ7 PASTOPRO.2D
T Ø15 T.PASTOPRO.O
B ØØ2 PASTOPRO.O
A ØØ2 F-
T Ø14 T.CONVERT
B ØØ2 CONVERT
A ØØ2 G-
T Ø1Ø T.VORLESER
B ØØ3 VORLESER
```

Sammeldisk #8

```
A ØØ2 PEEKER == HEFT 8/85 ==
A ØØ2 A-
A Ø23 HELLO
B 12Ø ASMDIV
A ØØ2 B-
A Ø41 DISKTEST
T ØØ2 DISKTEST.START
T Ø27 T.DISKTEST
A ØØ2 C-
A ØØ8 KOPY
A ØØ8 BATCHKOPY
T Ø05 T.GETSETINFO
B ØØ2 GETSETINFO
A ØØ2 BILDTEST
A ØØ2 D-
T Ø2Ø T.BOX.COPY
B ØØ3 BOX.COPY
T ØØ3 T.HSCRN
B ØØ2 HSCRN
B ØØ6 GRAF.QUATTRO.1
A ØØ2 E-
T Ø33 T.DOUBLE.LORES
B ØØ4 DOUBLE.LORES
A ØØ3 DOUBLE.LORES.DEMO
A ØØ2 F-
T ØØ7 AD-START.CMD
T Ø13 HMENUE.CMD
T ØØ3 AUFNAHME.CMD
T Ø11 AUFMASKE.CMD
T Ø11 AUSMASKE.CMD
T ØØ4 EDITFNAM.CMD
T ØØ4 SUCHFNAM.CMD
T ØØ6 SUCHVNAM.CMD
T ØØ5 SUCHBEME.CMD
T Ø14 SCHREIBA.CMD
T Ø15 SCHREIBL.CMD
T ØØ7 LOESCH.CMD
T ØØ3 SUCH.CMD
A ØØ2 G-
T Ø15 IDSEARCH
A ØØ2 H-
A ØØ4 FAKULTAET.DEMO
T Ø17 T.FAKULTAET
B ØØ4 FAKULTAET
A ØØ2 I-
A Ø11 GRAFIK.DEMOS
A ØØ2 J-
A ØØ2 ZEICHENJAGD
A ØØ2 K-
T Ø25 T.RAM.FRE.NEU
B ØØ3 RAM.FRE.NEU
```

Sammeldisk #9

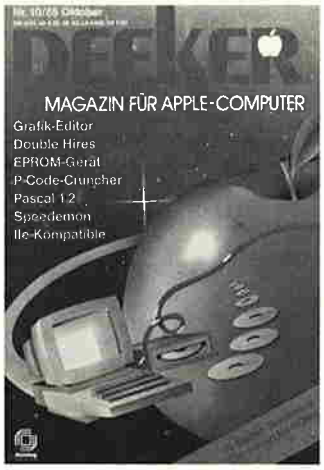
UCSD-Pascal-Diskette

```
DUPDIR.TEXT
DUPDIR.CODE
GETDOS.TEXT
GETDOS.CODE
MOUSESTUFF.TEXT
MOUSESTUFF.CODE
TESTMOUSE.TEXT
TESTMOUSE.CODE
DRAWMOUSE.TEXT
DRAWMOUSE.CODE
MOUSE.ASS.TEXT
MOUSE.ASS.CODE
MOUSE.LIBRARY
TRANSCEND.TEXT
TRANSCEND.CODE
RAMDISK94.TEXT
INIT.TEXT
INSTALL.TEXT
RAMDISK94.CODE
IDSEARCH.TEXT
IDSEARCH.CODE
CRUNCH.TEXT
CRUNCH.CODE
MOVER.TEXT
MOVER.CODE
```

```
CRUNCHER.TEXT
CRUNCHER.CODE
ALLG.TEXT
ALLG.CODE
MUEL.TEXT
MUEL.CODE
```

Sammeldisk #10

```
A ØØ2 PEEKER == HEFT 9/85 ==
A ØØ2 A-
T Ø46 T.AS.FILER
B ØØ4 AS.FILER
A ØØ4 STARTUP
A ØØ8 ENDUP
B ØØ4 RAMDISKLC
A ØØ2 B-
A ØØ5 MKBOOT.BAS
T Ø11 T.MKBOOT.OBJ
B ØØ2 MKBOOT.OBJ
T Ø12 T.DOBOOT.OBJ
B ØØ2 DOBOOT.OBJ
A ØØ2 C-
A ØØ8 SUPER.HGR
T ØØ3 SUPER.HGR.ASM
A ØØ2 D-
T Ø33 T.FLAG.MONITOR
B ØØ5 FLAG.MONITOR
T ØØ6 T.FLAG.MONITOR.TEST
B ØØ2 FLAG.MONITOR.TEST
A ØØ3 VERSAL
T ØØ4 T.VERSAL
```




```
A ØØ2 PEEKER == HEFT 1Ø/85 ==
A ØØ2 E-
A Ø24 GRAFIK.EDITOR
B ØØ6 GRAF.QUATTRO.1
A ØØ2 F-
A ØØ4 AS.DOUBLE.HIRES.DEMO
T Ø23 T.AS.DOUBLE.HIRES
B ØØ4 AS.DOUBLE.HIRES
A ØØ2 G-
T ØØ6 T.CHARSET
B ØØ2 CHARSET
A ØØ7 DEMO.CTRL
A ØØ3 CTRL.DISABLE
T ØØ2 T.ENABLE.DISABLE
A ØØ3 HTAB1.BUG
A ØØ3 INVERSE.BUG
T Ø1Ø T.INT.IIE.NEU
A ØØ2 H-
A Ø15 EPROM
T Ø16 T.EPROMMER
B ØØ3 EPROMMER
A ØØ2 I-
T ØØ5 CRUNCH.TEXT
T Ø29 MOVER.TEXT
T Ø55 CRUNCHER.TEXT
A ØØ2 J-
A ØØ8 PUZZLE
A ØØ8 PUZZLE.PDL
A ØØ2 K-
T Ø64 UCSD.TEXT
T Ø4Ø TURB.PAS
```

Sammeldisk #11

```
A ØØ2 PEEKER == HEFT 11/85 ==
A ØØ2 A-
A Ø29 SPRITE.EDIT
T ØØ6 T.SPRIT.EDIT.ASS
B ØØ2 SPRITE.EDIT.ASS
T Ø4Ø T.ASTA
B ØØ6 ASTA
T Ø11 T.ASTA.DEMO
B ØØ2 ASTA.DEMO
A ØØ2 ASTA.DEMO.1
B ØØ7 AD1
B ØØ2 AD2
B ØØ2 AD3
B ØØ2 AD4
B ØØ3 AD5
A ØØ2 B-
T Ø14 T.PLAKAT
A ØØ3 PLAKAT.INSTALL
B ØØ8 PLAKAT
A ØØ4 PLAKAT.DEMO
A ØØ2 C-
A Ø12 DISK.CHIRURG
A ØØ2 D-
T Ø31 FILER.Ø
B ØØ4 FILER.1
B ØØ2 FILER.2
T Ø37 SYSTEM.PATCH.Ø
B Ø11 SYSTEM.PATCH.1
T Ø19 PRODOS.Ø
B ØØ2 PRODOS.1
B ØØ2 PRODOS.2
B ØØ2 PRODOS.3
B ØØ2 PRODOS.4
A ØØ2 E-
T Ø26 T.PRINT.USING
B ØØ2 PRINT.USING
T Ø26 T.PRINT.USING.G
B ØØ2 PRINT.USING.G
A ØØ5 PRINT.USING.DATA
A ØØ5 PRINT.USING.G.DATA
A ØØ5 PRINT.USING.DEMO
A ØØ2 F-
T ØØ9 T.REF.TEST
B ØØ2 REF.TEST
```

Sammeldisk #12

```
A ØØ2 PEEKER == HEFT 12/85 ==
A ØØ2 A-
A ØØ3 COSMO CRUMBLE
T Ø52 T.CC2
T ØØ3 T.CC3
T ØØ9 T.CC4
T Ø2Ø T.CC5
T Ø18 T.CC6
T Ø81 T.CC8
B Ø19 CC2
B ØØ2 CC3
B ØØ3 CC4
B ØØ5 CC5
B ØØ5 CC6
B Ø34 CC7
B Ø18 CC8
B Ø28 CC.LEVELS
A ØØ2 B-
A Ø21 IW.DEMO
T ØØ2 T.IW.IN
B ØØ2 IW.IN
T ØØ5 T.IW.OUT
B ØØ2 IW.OUT
A ØØ2 ECHO
A ØØ2 C-
A Ø13 ASCII.EDITOR
A ØØ3 COPY.TEXT.DEMO
T ØØ8 T.COPY.TEXT
B ØØ2 COPY.TEXT
B ØØ9 ASCII.CODES
A ØØ2 D-
A ØØ3 GETTEXT.DEMO
T Ø19 T.GETTEXT
B ØØ3 GETTEXT
B ØØ3 GETTEXT.PRODOS
A ØØ2 E-
T Ø18 RAMDISK.PAS
A ØØ2 F-
B Ø88 JAHRESINHALT
A ØØ2 WORT.SUCHER
B ØØ3 WORT.SUCHER.OBJ
T ØØ6 DUMP.TEXT
```

Sammeldisk #13

```
A ØØ2 PEEKER == HEFT Ø1/86 ==
A ØØ2 A-
A Ø12 QUICK.RANDOM
A ØØ7 QUICK.SPEZIAL
A ØØ5 QUICK.TASC
A ØØ6 QUICK.DISK
A ØØ2 B-
A ØØ2 QUICKSORT.DEMO
T Ø3Ø T.QUICKSORT
B ØØ3 QUICKSORT
A ØØ2 C-
A Ø97 VOK.TRAINER
A ØØ9 VOK.COPY
B ØØ7 VOK.PACK
B ØØ5 VOK.BCOPY
B ØØ9 GWS.INFO
T 162 GWS.VOK
A ØØ2 D-
A ØØ4 AGE.DEMO
T Ø56 T.AGE
B ØØ6 AGE
A ØØ2 E-
A Ø18 GRAFIK.DEMOS.2
```

Sammeldisk #14

```
A ØØ2 PEEKER == REST Ø1/86 ==
A ØØ2 A-
T Ø57 T.PROTODOS
B ØØ8 PROTODOS
A ØØ2 B-
T Ø4Ø DESIGNER.TEXT
A ØØ2 C-
T Ø36 READPAS.PAS
A ØØ2 PEEKER == HEFT Ø2/86 ==
A ØØ2 D-
A ØØ2 KOMPLEMENT.DEMO
T ØØ4 T.KOMPLEMENT
B ØØ2 KOMPLEMENT
A ØØ2 E-
A ØØ3 DOSMOVER.START
T Ø49 T.DOSMOVER
B ØØ5 DOSMOVER
A ØØ2 F-
A ØØ8 MATRIX
A ØØ7 VEKTOR
A ØØ5 AMPERSOFT1
A ØØ9 AMPERSOFT2
A ØØ4 AMPERSOFT3
A ØØ4 AMPERSOFT4
A ØØ3 AMPERSOFT5
A ØØ7 AMPERSOFT6
A ØØ4 AMPERSOFT7
A ØØ4 AMPERSOFT8
A Ø14 LRS1
A ØØ7 LRS2
A ØØ2 G-
A Ø15 INIT.SERIELL
A ØØ2 H-
T ØØ9 T.CHRGET.SPRUNG
B ØØ2 CHRGET.SPRUNG
A ØØ2 I-
A ØØ2 SC.MOVER.DEMO
T Ø14 T.SCREEN.MOVER
B ØØ2 SCREEN.MOVER
A ØØ2 J-
A ØØ7 DARSTELLUNG.3D
A ØØ2 K-
T Ø29 TIPS5.TEXT
A ØØ2 L-
```

Sammeldisk #15

```
A ØØ2 PEEKER == HEFT Ø3/86 ==
A ØØ2 A-
A ØØ3 REELL.1
A ØØ3 REELL.1A
A ØØ3 REELL.2
A ØØ3 REELL.3
A ØØ3 REELL.4
A ØØ4 REELL.5
A Ø12 REELL.6
A ØØ3 REELL.7
A ØØ5 REELL.8
A ØØ3 REELL.9
A ØØ6 REELL.1Ø
A ØØ2 B-
T Ø51 T.MAKESUB
B ØØ7 MAKESUB
A ØØ2 C-
T Ø15 DATENDEMOS
T ØØ9 SUBDEMOS
T Ø37 STRINGDEMOS
T Ø11 FILEDEMOS
A ØØ2 D-
T ØØ5 FILETEST.TEXT
T Ø35 FIBTEST.TEXT
A ØØ2 E-
A ØØ3 MESSWERT.START
A Ø54 MESSWERT
B ØØ6 MESSWERT.TITEL
B ØØ5 MW.ZEICHENSATZ
```

Sammeldisk #16

```
A ØØ2 PEEKER == HEFT Ø4/86 ==
A ØØ2 A-
B Ø27 MACROEDITOR
A ØØ2 B-
A ØØ2 M8TEST
T ØØ4 T.M8RLN
B ØØ2 M8RLN
T ØØ4 T.M8LRN
B ØØ2 M8LRN
T ØØ4 T.M8RLT
B ØØ2 M8RLT
T ØØ4 T.M8LRT
B ØØ2 M8LRT
A ØØ3 M16TEST
T ØØ5 T.M16RLN
B ØØ2 M16RLN
T ØØ5 T.M16LRN
B ØØ2 M16LRN
T ØØ4 T.M16RLT
B ØØ2 M16RLT
T ØØ4 T.M16LRT
B ØØ2 M16LRT
A ØØ2 MULT8RLN.DEMO
T ØØ8 T.MULT8RLN
B ØØ2 MULT8RLN
A ØØ3 MULT16RLN.DEMO
T ØØ8 T.MULT16RLN
B ØØ2 MULT16RLN
A ØØ3 D16TEST
T ØØ5 T.D16V1
B ØØ2 D16V1
T ØØ4 T.D16V2
B ØØ2 D16V2
T ØØ4 T.D16V3
B ØØ2 D16V3
T ØØ4 T.D16V4
B ØØ2 D16V4
A ØØ3 D24TEST
T ØØ4 T.D24V2
B ØØ2 D24V2
A ØØ3 DIV16.DEMO
T ØØ7 T.DIV16
B ØØ2 DIV16
A ØØ3 DIV24.DEMO
T ØØ8 T.DIV24
B ØØ2 DIV24
A ØØ2 C-
A Ø15 FONT.LOADER
T Ø23 T.FONT.LOADER.OBJ
B ØØ3 FONT.LOADER.OBJ
T Ø12 T.FONT.CONVERT
B ØØ2 FONT.CONVERT
B ØØ5 ASCII.FONT
B ØØ5 DEUTSCH.FONT
B ØØ5 SONDERZEICHEN.FONT
A ØØ2 D-
A Ø38 HAUSHALT
A Ø38 HAUSHALT.4Ø
T ØØ2 DATEI
A ØØ2 E-
A ØØ4 DUMP8ØREL.DATA
T ØØ5 T.DUMP8ØREL
B ØØ2 DUMP8ØREL
A ØØ2 F-
T Ø37 KYANASM
T Ø2Ø HEXTYPEN
A ØØ2 G-
T ØØ5 FUNKSTART.TEXT
T ØØ3 FUNKDEMO.TEXT
T ØØ2 FUNKUNIT.TEXT
T ØØ2 FUNKEXEC.TEXT
A ØØ2 H-
A Ø15 MUSIK.EDITOR
T Ø1Ø T.TONROUTINE
B ØØ2 TONROUTINE
B Ø18 M.SONATA
A ØØ2 SONATA
```

Sammeldisk #17

```
A ØØ2 PEEKER == HEFT Ø5/86 ==
A ØØ2 A-
T Ø61 WM86.PAS
T Ø11 VORSPANN.PAS
T Ø25 ENTSCHEI.PAS
A ØØ2 B-
T Ø46 PRINT.ASS.TEXT
T ØØ4 FASTPRINT.TEXT
T ØØ5 DEMOPRINT.TEXT
A ØØ2 C-
T Ø33 LISTCODEF.TEXT
T ØØ3 LEVELØ.TEXT
T ØØ4 ASSEM.TEXT
A ØØ2 D-
A Ø12 PI.START
T Ø37 T.PI
B ØØ5 PI
A ØØ2 E-
T ØØ2 PATCH.STARTER
A ØØ7 SQ.PATCHER
```

Sammeldisk #18

DOS-Teil

```
A ØØ2 PEEKER == HEFT Ø5/86 ==
A ØØ2 A-
B Ø22 EDIT
A ØØ2 B-
A ØØ3 MDB.KOPY.SPEZIAL
A ØØ4 MDB.KOPY
T Ø16 T.MDB.KOPY.OBJ
B ØØ3 MDB.KOPY.OBJ
A ØØ2 PEEKER == HEFT Ø6/86 ==
A ØØ2 C-
T ØØ2 PLOT.3.PRO
A Ø59 PLOT.3.E
B ØØ7 PLOT.BX
T Ø55 T.PLOT.BX
T Ø15 PLOT.HELP.1
T Ø12 PLOT.HELP.2
T Ø11 PLOT.HELP.3
T Ø11 PLOT.HELP.4
T ØØ8 PLOT.HELP.5
T ØØ9 PLOT.PATCH
T ØØ5 SUPERDUMP.PATCH
A ØØ2 D-
A Ø11 PRODOS.BACKUP
T Ø24 T.PRO.BACKUP.0
B ØØ4 PRO.BACKUP.0
A ØØ5 KOPY.16Ø.SPUR
T Ø18 T.KOPY.16Ø.SPUR.0
B ØØ3 KOPY.16Ø.SPUR.0
```

USCD-Pascal-Teil

CALLDUMP.TEXT
CALLDUMP2.TEXT
SUPERDUMP.TEXT
SUPERDUMP.LIB
EPSON.TEXT
EPSON.CODE
IMAGEWRITR.TEXT
IMAGEWRITR.CODE
LIB.TEXT

Sammeldisk #19

A 002 PEEKER == HEFT 07/86 ==
A 002 A-
A 002 STIEHL
A 003 HELLO
B 006 DDMOVER
B 050 REGISTER.UTILITIES
B 024 REGISTER.RUNTIME
B 009 REGISTERSTARTER.OBJ
B 030 REGISTERREDIGIERER.OBJ
B 018 REGISTERSORTIERER.OBJ
B 019 REGISTERMISCHER.OBJ
B 020 REGISTERDRUCKER.OBJ
B 018 REGISTERUMWANDLER.OBJ
A 007 REGISTERTESTER
A 008 REGISTERUMDREHER
A 007 REGISTERTEILER
A 002 B-
A 005 MOUSORY
A 013 MOUSORY.DEMO
A 019 MOUSORY.GAME
B 002 MOUSORY.BELL
B 024 MOUSORY.SET1
B 022 MOUSORY.SET2
B 026 MOUSORY.SET3
B 034 MOUSORY.BILD
A 002 C-
A 002 START
A 005 HP
A 003 UP1
A 003 UP2
T 007 T.RUN.FILE
B 002 RUN.FILE
A 002 D-
A 015 CLOCK
T 020 CLOCK.EXEC
A 002 E-
A 006 PRODOS.LIB.DEMO
T 074 T.PRODOS.LIB
B 009 PRODOS.LIB

Sammeldisk #20

Pic-Edit-UCSD-Pascal-Diskette

SYSTEM.APPLE (Dummy-Files!)
SYSTEM.PASCAL (Betriebssystem)
SYSTEM.MISCINFO (müssen Sie selbst)
SYSTEM.CHARSET (überspielen.)
SYSTEM.ATTACH
SYSTEM.STARTUP
SYSTEM.LIBRARY
CRUNCHER.CODE
ASCII.FONT
GERMAN.FONT
MATH.FONT
GREEK.FONT
SUPER.SUB.FONT
MENU.GRAF
PRINTER.INFO
EPSON
IMAGEWRITER
ARROWS.KEYS
DESIGNER.CODE

Sammeldisk #21

A 002 PEEKER == HEFT 08/86 ==
A 002 A-
A 005 LOGIK
T 004 MLOGIK
A 002 B-
T 019 T.NETZWERK
B 003 NETZWERK
A 002 C-
T 006 T.UNIFORMAT
B 002 UNIFORMAT
A 002 PEEKER == HEFT 09/86 ==
A 002 D-
A 003 HARDBREAKER
A 006 GEN.EXEC
T 009 T.NMI
B 003 NMI
T 007 T.LCMOVE
B 002 LCMOVE
T 003 HARDBREAKER.INIT
T 003 HARDBREAKER.INIT.BASIS
T 002 SAVEMEM
A 002 E-
T 052 FILER.PAS
A 002 F-
T 026 DREIECK.TEXT
A 002 G-
T 024 STRINGUTILS.I
T 013 STRINGDEMOS.P
A 002 H-
A 002 EINZEL.TASC
B 021 EINZEL.RUNTIME
B 061 EINZEL.OBJ
A 056 EINZEL.KALK
T 002 EINZEL.KONST
T 006 EINZEL.KLEIN
T 006 EINZEL.GROSS
A 002 I-
A 003 FUNKTIONEN START
A 011 FUNKTIONEN ANLEITUNG
B 002 FUNKTIONEN EXC
A 057 FUNKTIONEN

Sammeldisk #22

A 002 PEEKER == HEFT 10/86 ==
A 002 A-
A 009 SPURJUSTAGE.SIMULATION
A 003 MOTORTEST
A 002 B-
T 084 T.CHANGE.REMS
B 008 CHANGE.REMS
T 089 T.CHANGE.REMS.DOS.3.3
B 008 DOS.CHANGE.DOS.3.3
A 002 C-
A 009 HEX.MONITOR
A 002 D-
A 004 DHRCG.LOADER
T 043 T.DHRCG
B 006 DHRCG
B 005 GERMAN.SET
A 003 AMPER.DOUBLE.HIRES.BAS
B 004 AMPER.DOUBLE.HIRES
A 002 E-
T 062 DISK1.TEXT
T 078 DISK2.TEXT
T 079 DISK3.TEXT
A 002 F-
T 009 T.PRODOS.PATCH.111
B 003 PRODOS.PATCH.111

Sammeldisk #23

A 002 PEEKER == HEFT 11/86 ==
A 002 A-
T 120 T.DISK.MONITOR
B 016 DISK.MONITOR
A 002 B-
T 019 T.DISK.SUCHER
B 003 DISK.SUCHER
A 002 C-
T 022 T.STRCAT
B 003 STRCAT
A 003 STRCAT.DEMO1
A 006 STRCAT.DEMO2
A 002 D-
T 051 UTILITY.TEXT
T 028 UTIL.ASS.TEXT
T 004 DEMO.TEXT
A 002 E-
T 066 PRINT.FILE.TEXT
A 002 F-
A 003 DEMO.INPUT.FORM
T 008 T.INPUT.FORM
B 002 INPUT.FORM
A 002 G-
A 002 WORT.SUCHER
T 011 T.WORT.SUCHER.OBJ
B 003 WORT.SUCHER.OBJ
T 134 JAHRESINHALT
A 002 H-
A 002 PATCH
B 003 LOCH

Sammeldisk #24

A 002 PEEKER == HEFT 12/86 ==
A 002 A-
T 062 T.FILECOPY
B 007 FILECOPY
A 002 B-
A 008 STARTUP
A 002 C-
A 003 MACRO.BIN.MAKER
A 002 START.FW.MACRO
T 008 MACRO.TXT
B 002 MACRO.BIN
T 004 TESTTEXT
T 014 T.DRIVER
B 003 DRIVER
A 002 D-
A 021 BASIC.MASKE
A 002 E-
A 005 FMULT.DEMO
T 006 T.FMULT.PATCH
B 002 FMULT.PATCH
A 002 F-
T 017 T.LINE.REFS
B 003 LINE.REFS
A 002 LINE.REFS.DEMO
A 002 G-
T 098 T.DHGR.IIPLUS
B 009 DHGR.IIPLUS
B 009 DHGR.IIPLUS.PRO
B 009 DHGR.IIPLUS.48K
A 004 DHGR.GENAU
A 004 DHGR.EPSON
A 002 H-
T 067 LIBRARY1.TEXT
T 060 LIBRARY2.TEXT
A 002 I-
T 006 AUFSPALTEN.TEXT
A 002 J-
A 002 START.FW.LQ800
T 008 MACRO.LQ800.TXT
B 002 MACRO.LQ800.BIN
T 004 TESTTEXT.LQ800
A 002 START.FW.FX80
T 006 MACRO.FX80.TXT
B 002 MACRO.FX80.BIN
T 003 TESTTEXT.FX80
A 002 START.FW.IMAGE
T 006 MACRO.IMAGE.TXT
B 002 MACRO.IMAGE.BIN
T 003 TESTTEXT.IMAGE

Sammeldisk #25

A 002 PEEKER == HEFT 1/87 ==
A 002 A-
A 025 TABLE.EDITOR
A 028 SHAPE.EDITOR
A 027 SHAPE.CREATE
B 002 ST.EDIT
A 003 UEBERSICHT
A 002 B-
A 011 TRACE.KORREKTUR
A 002 C-
A 017 PROGRAMMVERWALT
A 005 ALPHASORT
T 002 PROGRAMMDATEI
A 002 D-
A 007 ABU.DEMO
T 074 T.ABU
B 013 ABU
A 002 E-
T 007 CAT.P
T 060 CAT.I
A 002 F-
T 018 DISM1.TEXT
T 069 DISM2.TEXT
T 078 DISM3.TEXT
T 046 DISM4.TEXT
A 002 G-
A 014 BAUER
B 004 BAUER.OBJ
T 003 BAUER.EXEC

Sammeldisk #26

A 002 PEEKER == HEFT 2/87 ==
A 002 A-
A 003 TEXT.TEST
B 002 TEXT.LQ800
T 004 T.TEXT.LQ800
A 003 HGR1.TEST
B 002 LQ800.HGR1
T 009 T.LQ800.HGR1
A 003 HGR3.TEST
B 002 LQ800.HGR3
T 010 T.LQ800.HGR3
A 003 HGR3Q.TEST
B 002 LQ800.HGR3Q
T 010 T.LQ800.HGR3Q
A 002 B-
T 011 READIN.TEXT
A 002 C-
T 006 V80.TEXT
A 002 D-
T 003 DEMO.TEXT
T 020 LONGINT.TEXT
A 002 E-
A 035 FX80.PATCHER
T 006 T.FX80.INIT
B 002 FX80.INIT
A 002 F-
T 014 T.BRK
B 002 BRK
T 010 T.BRK.TEST
B 002 BRK.TEST
A 002 G-
A 005 STIEHL
B 030 DB-MEISTER.OBJ
B 011 DB-STARTER.OBJ
B 036 DB-PFLEGER.OBJ
B 002 DB-BILDSCHIRM
A 002 H-
B 077 ASM.SYSTEM
B 002 ASM.GP
A 002 I-
A 003 MATCH.TEST
T 033 T.MATCH
B 004 MATCH
A 002 J-
T 016 T.OHO.RAMDISK
B 003 OHO.RAMDISK
T 016 T.OHO.RAMDISK.ACC
B 003 OHO.RAMDISK.ACC
B 003 OHO.RAMDISK.256
B 003 OHO.RD.ACC.256
A 005 RENAME.VOL.RD
A 002 K-
T 072 INHALT.DISKETTEN

Sammeldisk #27

A 002 PEEKER == HEFT 3/87 ==
A 002 A-
B 019 DB-SORTER.OBJ
B 010 DB-SORTER2.OBJ
B 028 DB-FILTER.OBJ
B 049 DB-DRUCKER.OBJ
B 002 DB-PRINTER
T 002 LEERKOPF
A 002 B-
T 002 MENUE.DBLH.START
A 006 MENUE.DBLH
A 017 BLOCKEDIT.DBLH
A 020 BLOCKLINK.DBLH
A 016 CHAREDIT.DBLH
A 018 CONVERT.DBLH
A 009 STARTER.DBLH
B 005 DEUTSCH.SET
B 005 BYTE.SET
B 020 COMP.BLK
B 020 GRAF.BLK
B 004 DBLHLD
B 024 DBLHTR
B 002 CONVERT
T 010 T.DBLHLD
T 120 T.DBLHTR
T 004 T.CONVERT
T 002 DEMO.START
A 009 DEMO
B 038 DEMO.TBL
A 002 C-
T 045 INHALT.AB.10.86

Sammeldisk #28

DB-Meister-Quelltexte

Aktuelle Computerbücher

1986, 246 S., DM 48,–
ISBN 3-7785-1379-6

Begleitdiskette: DM 44,–
ISBN 3-7785-1380-X

Das Buch beschreibt die Assembler-Programmierung im Rahmen des CP/M-Betriebssystems. Es wurde vorrangig für die Besitzer von Apple II-Geräten konzipiert. Der überwiegende Teil des Buches ist aber geräteunabhängig und somit auch für die Benutzer anderer CP/M-tüchtiger Fabrikate (mit 8080/Z80-Prozessor) von Interesse.
Einsteiger erhalten im ersten Abschnitt einen Intensiv-Kurs (auch in der DDT-Anwendung). Für Kenner der 6502-Programmierung werden viele Querverweise vorgestellt. Der Z80-Befehlsvorrat wird im Detail erläutert. Es folgen wichtige Basisprogramme (Konvertierungen, Binär-und BCD-Arithmetik), umfassende Diskussionen der Debugger, Assembler und Linker sowie des CP/M-Betriebssystems (Speicherverwaltung, BDOS und BIOS-Aufrufe, Bildschirmfunktionen). Den Abschluß bilden lauffähige Beispielprogramme und Utilities und ein umfangreicher Tabellen-Anhang.

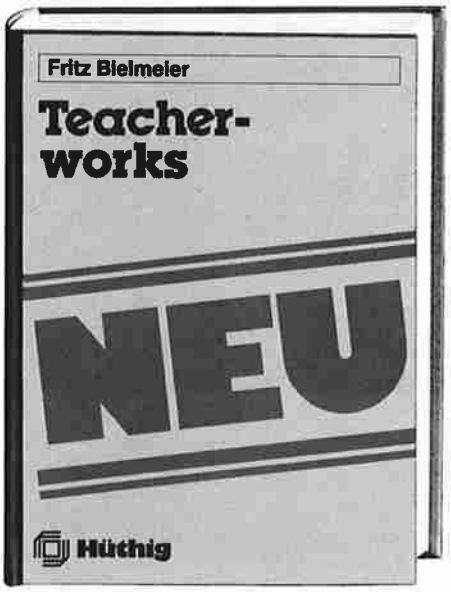
1986, 216 S., kart., DM 42,–
ISBN 3-7785-1246-3

Endlich einmal ein Buch für den fortgeschrittenen Programmierer, das mit den besonderen Aspekten und speziellen Möglichkeiten von PASCAL vertraut macht. Dies geschieht unter besonderer Berücksichtigung von Apple-Pascal, der Implementation von UCSD-PASCAL auf dem Apple II.
Im ersten Teil werden die hervorragenden grafischen Möglichkeiten, welche die Unit TURTLEGRAFICS bietet, dargestellt. Der zweite Teil geht ausführlich auf die schon fast professionellen Ein-/Ausgabemöglichkeiten dieser Sprache ein und erklärt unter anderem die Konvertierung von DOS-Files in UCSD-Files.
In einem dritten Teil wird die automatische Bedienung des Systems behandelt. Das UNIT-Konzept und die Modularisierung von Programmen bilden mit den erforderlichen Programmiertechniken den vierten Teil.
Zahlreiche kommentierte Beispielprogramme, die das Erläuterte veranschaulichen und zusätzlich sinnvolle Utilities für eigene Anwendungen bilden, runden zusammen mit deutschsprachigen Anhängen den Band ab.

Teacherworks: Notenverwaltung und Korrekturprogramme für Lehrer

von Fritz Bielmeier
1987, Programm und Anleitung, DM 98,–
ISBN 3-7785-1464-4

Teacherworks wurde von einem Studienrat aus den Anforderungen der Praxis heraus entwickelt. Es erledigt wirklich ökonomisch und absolut zuverlässig alle lästigen Büroarbeiten bei Korrektur und Notenfindung, wobei es dem Wunsch des Benutzers nach Einfachheit, Komfort und beachtlicher Schnelligkeit gerecht wird und strenge Anforderungen an die Sicherheit der Daten erfüllt.
Teacherworks ist unabhängig von Schulgattungen nutzbar und anpassungsfähig bezüglich der Druckformulare.
Das Korrekturprogramm benutzt für jede schriftliche Prüfungsarbeit Listen der Schülernamen und der erzielten Punkteleistung der Schüler, einen Notenschlüssel und einen Vermerk über die erreichbaren Rohpunkte der Teilaufgaben. Die Notenschlüssel können individuell erstellt oder aus einer Daten-Bibliothek entnommen werden. Der Endausdruck enthält alle Korrekturdaten, wie sie von der Schulleitung und Fachbetreuung gefordert werden, unter anderem auch Statistiken. Die Korrekturergebnisse (Note, Datum, Art) werden automatisch in die Notenliste übernommen. Zu dieser können Sie im Notenprogramm mündliche Noten aller Art hinzufügen, die sofort im Durchschnitt verrechnet werden. Die Gewichtung der einzelnen Noten ist frei veränderlich. Notenlisten können je Fach und Klasse oder Schüler ausgedruckt werden, z. B. zur Zeugnisvorbereitung und vor Elternsprechtagen.
Systemvoraussetzung: Apple IIc/e mit 80 Zeichenkarte, ProDOS, Imagewriter oder Epson-Drucker.



☐ H. Kersten, **Apple-CP/M: Assembler-Programmierung,** ISBN 3-7785-1379-6, DM 48,–

☐ Begleitdiskette, ISBN 3-7785-1380-X, DM 44,–

☐ Juhnke und Redlin, **Apple PASCAL für Fortgeschrittene,** ISBN 3-7785-1246-3, DM 42,–

☐ F. Bielmeier, **Teacherworks,** ISBN 3-7785-1464-4, DM 98,–

BESTELLCOUPON

Gewünschte Bücher bitte ankreuzen und an Dr. Alfred Hüthig Verlag, Postfach 102869, 6900 Heidelberg, schicken.

Name _____

Straße _____

Ort _____

Datum _____ Unterschrift _____

Hüthig

Warum wollen Sie ein teures Textverarbeitungsprogramm kaufen, wenn es ein billiges und besseres gibt?

Fast-Writer

von Harald Grumser
Programmdiskette und Handbuch
Gerätevoraussetzung: Apple IIe oder IIc (nicht II+)
DOS-3.3-Version.
Normalpreis DM 128,– (ISBN 3-7785-1419-9)
Sonderpreis für Peeker-Abonnenten DM 98,–

ProDOS-Version.
Normalpreis DM 128,– (ISBN 3-7785-1421-0)
Sonderpreis für Peeker-Abonnenten DM 98,–
Kombinationspreis für Bezieher der DOS-3.3-Version DM 28,–

Fast-Writer läuft auch auf neuem Apple GS!

Der Fast-Writer von Harald Grumser ist in zahlreichen Funktionen wie Scrollen, Suchen und Ersetzen mit Abstand das schnellste und damit angenehmste Textverarbeitungsprogramm für den Apple IIe oder IIc.

Flexibilität

Viele Textverarbeitungsprogramme sind geschützt und laufen deshalb nur in Verbindung mit normalen Disk-II-Laufwerken. Nicht so der Fast-Writer!

– Der Fast-Writer modifiziert weder DOS 3.3 (oder Diversi-DOS) noch ProDOS und kann deshalb mit BRUN FAST.WRITER gestartet werden. Unter Diversi-DOS ist der Fast-Writer dann in 3 Sekunden im Speicher. Vergleichen Sie einmal, wie lange es dauert, bis andere Textprogramme im Speicher sind!

– Da der DOS-3.3-Fast-Writer in den oberen 16K (= Language Card) liegt, kann man ihn vorübergehend verlassen und mit einem einfachen Befehl wieder starten. Mit anderen Worten: Der Fast-Writer ist permanent verfügbar, auch wenn Sie zwischendurch beispielsweise mit FID Dateien kopiert haben.

– Der Fast-Writer läuft mit allen externen Datenspeichern, die für DOS 3.3 oder ProDOS gedacht sind, z.B. mit dem Erphi-160-Spur-Subsystem, mit der Megaboard-MDB-Festplatte, mit RAM-Karten usw. Spezielle Anpassungen sind nicht erforderlich. Suchen Sie einmal ein Textprogramm, das mit diesen Datenspeichern auf Anhieb funktioniert!

– Der Fast-Writer kann mühelos über ein Menü für Ihre speziellen Aufgaben konfiguriert werden. Sie können z.B. per Knopfdruck die Zeilenbreite (normal 80 Zeichen) am Bildschirm einstellen, wobei ab einer Breite von weniger als 41 Zeichen automatisch auf die größere Bildschirmschrift umgestellt wird. Ferner können Sie die Größe des Arbeitsspeichers (insgesamt ca. 35 500 Zeichen) beliebig in Textspeicher und Hilfspuffer (für Löschen und Blockverschieben) aufteilen. Wenn Sie z.B. große Textblöcke im Speicher zu verschieben haben, so können Sie einen entsprechend großen Hilfspuffer von z.B. 10 000 Zeichen einrichten. Damit entfällt das zeitraubende Zwischenspeichern und Einlesen von Diskette.

Befehlsvorrat

Der Fast-Writer verfügt über eine große Zahl von Befehlen, von denen Sie in der Praxis jedoch nur wenige benötigen. Fünf Befehlsübersichten sind durch eingebaute Hilfsübersichten immer abrufbar, so daß Sie schon nach einer mehrstündigen Einarbeitung auf das Handbuch verzichten können. Eine Auswahl der wichtigsten Befehle:

– Freie Cursorbewegung in allen vier Richtungen mit eingebauter Schnell-Scroll-Routine.

– Diverse, per Knopfdruck ein- und ausschaltbare Optionen, z.B. Wortumbruch/kein Wortumbruch, Return sichtbar/unsichtbar, Kopfzeile (Statuszeile) mit Speicherbelegung, Cursorposition usw. eingeblendet/ausgeblendet, Bildschirm geteilt/ungeteilt, Tabulatorleiste sichtbar/unsichtbar, überschreibmodus (statt normalen Einfügmodus) ein/aus usw.

– Eingabe von Kontrollbuchstaben (einschließlich Ctrl-V!) möglich. Automatische Konvertierung in Groß- oder Kleinschreibung (unter Berücksichtigung der Umlaute und ß!)

– Extrem schnelles Suchen und Ersetzen von Zeichenketten (vorwärts und rückwärts).

– Makros frei definierbar und per Knopfdruck abrufbar. Makros können nicht nur stereotype Wortfolgen sein (z.B. „Sehr geehrte Herren"), sondern auch alle Befehlsfolgen, die man beim Fast-Writer sonst über die Tastatur eingeben würde. So läßt sich beispielsweise ein Text automatisch von Laufwerk 1 laden und auf Laufwerk 2 speichern.

– DOS-Kommandos wie Catalog, Delete, Rename usw. immer verfügbar (bei DOS 3.3 zusätzlich Init, bei ProDOS zusätzlich Online und Datum)

– Ausdruck auf Matrixdrucker (normal endlos), Schreibmaschine (normal mit Einzelblatt) und zu Kontrollzwekken auf Bildschirm; links- und rechtsbündig, zentriert und Blocksatz; einstellbarer linker, rechter und oberer Rand (im Text änderbar), bei Bedarf mit Kopfzeile und Paginierung usw. Der Ausdruck kann über eigene Druckertreiber umgelenkt werden, um z.B. Probleme mit Typenrädern, Steuerzeichen usw. zu beheben.

– Makros, Druckparameter, Druckertreiber und Tabulatoren können auf Diskette gespeichert werden.

Hüthig Software Service · Postfach 10 28 69 · 6900 Heidelberg 1